FIGHT FOR YOUR LIFE!

...in this gripping arcade adventure

Cram more into ram

Cure tape loading bugs

Protect your Break key

AMAZING ANIMATION

Krazy Kaleidoscope + Bouncing Ball

CONTENTS

electron user

Rush to upgrade machines

THE rush by users to upgrade their Electrons to 64k is causing problems for leading supplier Slogger.

The firm's master ram board upgrade has proved so popular that extra space has had to be found in order to store the large number of machines coming in for conversion.

Barry Johnson, sales manager of Slogger (0634 52303), told *Electron User*: "We are working a 25 hour day at the moment in order to get the backlog of work cleared. Our order book is almost full – but that's a problem we don't mind having.

"We are expecting further large orders for ram boards prior to the Christmas buying period".

The advantages of the board are that games speeds can be up by as much as 300 per cent, word processing is faster and there is a built-in turbo driver.

Slogger has brought out a new version of its ram board with a different operating system. This has the ability to run Elite using a joystick, which until now had not been possible on the Electron.

Bad weather boost for Electron games

THE wet start to the summer has helped boost UK entertainment software sales to unseasonally high levels, with Electron games suppliers in particular reaping the benefits.

Average total sales per shop have been much higher than normal, according to figures revealed by national surveys like the weekly Gallup retail round-up.

"The wettest June on record and the daily failure of temperatures to rise above 65 degrees are a possible benchmark indicating how many home computers are staying out of the cupboard this summer", says Gallup.

The boom in games software sales helped to put the Electron in fourth place recently – ahead of Commodore, BBC and Atari – according to Gallup's weekly survey of the share of sales by machine.

Electron users are buying in such quantities these days they are significantly affecting the success of games releases.

Says Gallup:"Nine months following its original launch, Elite found Paperboy once again at the top – this time of the Electron, Commodore and BBC charts.

"The reaction from the Electron-owning population seems to be the most responsive, with first week single format sales enough to command a high Top 20 individual placing and the title doing enough to make Electron the fourth largest market of the moment".

Recent weeks have seen Electron's share of machine-specific sales creep up behind those for the Spectrum, Commodore and Amstrad sectors which dominate the marketplace.

In terms of the percentage of titles sold, the Electron has been consistently in fifth place, just behind Commodore but ahead of BBC Micro and Atari.

Rhyming tutor

FIVE traditional children's favourites are colourfully featured on Nursery Rhymes, the latest educational program for the Electron from Database Software.

Aimed at children of nursery school and pre-school age, it uses full colour graphics and sound to make learning a fun activity.

Correct answers help the child complete the storyline of each rhyme, which is played via the micro's sound channel when the task has been successfully completed.

Jack and Jill, Hey Diddle Diddle, Hickory Dickory Dock, Humpty Dumpty and See Saw teach a variety of skills, including word recognition and basic spelling, dexterity and coordination, numbers and telling the time, and recognising shapes.

Nursery Rhymes costs £5.95 on tape and £6.95 on disc.

For more details, see Page 44.

NOT FOR SALE...

Advanced Computer Products, the Electron specialist, has produced a highly regarded new add on to extend its range. But far from considering it as peripheral, managing director John Huddlestone has made it the centre of attention.

The product is the result of nine months development by John's wife, Jane, who plays a major part in the business, though he provided the initial programming.

It is expected that Sophie Jane, the third in their series of offspring, will be promoted strongly.

Bingo! It's an Electron

AN Electron is playing a key role in the Alhambra Bingo Club in the West Midlands.

It has been brought in to replace the traditional revolving drum and numbered balls.

By running a program of random selection, it provides the numbers to be called out.

"We've kitted it out with flashing lights and the old ladies love it", John Rush, the club's manager, told *Electron User*.

KNIT ONE, PUSH ONE, MICRO ONE

READERS of *Electron User* have been bitten by the knitting bug thanks to a new suite of programs that went on sale last month.

Knitwear Designer encourages people to produce their own money-saving sweater patterns on the Electron.

And it has had a big impact on knitting enthusiasts – even in the Shetland Islands, traditional home of top quality sweater production.

"Yes, we are famous up here for our knitwear", said Jill Deas who lives on the Isle of Harris, "but perhaps this computer program can teach us a thing or two".

Many people who bought the package have written or phoned to say how much more it can do with a touch of imagination.

"You don't have to restrict yourself to jumpers", said one. "I've designed some cardigans and sleeveless pullovers with Knitwear Designer – and even some woollies for my children's teddy bears".

A reader phoned to say: "I've been able to save a lot of money by using the freehand program to produce knitwear in my youngsters' school colours...a lot cheaper than anything in the shops".

Details of how to obtain the five-program suite are on Page 61.

Bouncing with the best!
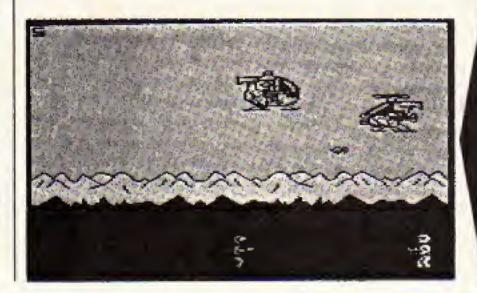
THE Electron is mixing with some high class company in the computer world these days. It can be found rubbing shoulders with the likes of Commodore Amigas, Atari STs and even the sensational Archimedes. The reason for this is that it has joined the bouncing ball club.

The back room boys at Commodore first devised the technique to demonstrate the graphics and sound capabilities of the Amiga. It involved an animated ball bouncing round the monitor screen. So effective did it prove that Atari offered its own version when it introduced the ST.

And even Acorn adopted the demonstration when it recently unveiled the fastest micro in the world, the Archimedes.

Not to be outdone, the *Electron User* has got in on the act on behalf of its readers. So if you want to bounce a ball the way that owners of far more expensive machines have been doing of late,

FOLLOWING an outcry from Electron users, Superior Software has converted Stryker's Run I and II for the machine.

The game, which was a big success on the BBC Micro, follows the plight of commander John Stryker as he battles to get a message to his companions on the far side of a battlefield.

Price £9.95 on cassette. Stryker Run II will also be available on 3.5in disc for £14.95.

Days of valves and Agol

Another claimant for the title of Britain's oldest Electron enthusiast has come forward.

Last month's Electron User featured a story about 73-year-old Ron Panting, who took up computing in his retirement so he could share his granddaughters' hobby.

But Ron was still in nappies when Allan Stevens of St Andrews, Fife, started school and began the life-long interest in mathematics which eventually led him to Electron computing.

Now 78, Allan spent the last 20 years of his working life as a medical records officer and can remember seeing his first computer in Edinburgh more than 15 years ago.

"I recall the language it used was Algol and it had enough valves to heat a building – I couldn't see a future for it in medical records at that time", he told Electron User.

"But it must have made an impression on me because some time after I retired in 1974 I heard about the ZX81 and got hold of one.

"This was my first hands-on experience and I must confess I was disappointed by the machine's limitations.

"I persevered with microcomputing, however, and became involved with other machines. Two years ago I bought my Electron – and I've been in love with it ever since.

"My equipment is the unadorned Electron – I can't afford any Plusses – and I'm definitely not interested in games, though I've been typing in the Patience listing.

"That's because there's a mathematical element, and maths and music are my great interests apart from the Electron. I only ever bought one commercial game and that was a boring wash-out".

Memory conserver

A 64k buffer which can be plugged in to any computer using a Centronics printer "and forgotten" is being sold in Britain by Frontier Software.

Manufactured by the Supra Corporation in the US, the MicroStuffer operates very much as a normal printer buffer, but it is a separate hardware device which leaves all the micro's memory free.

The product, connected between computer and printer, automatically buffers any data sent to the printer, freeing the computer for other work.

Frontier (0423 67140) says MicroStuffer includes a self test which checks the 64k ram on switch-on.

Sales manager Dave Fields said: "For the same price as most printer buffer programs, Microstuffer will work with just about any computer, can be left under your desk and forgotten about. It will also greatly increase your productivity".

Price £49.95.

Bat power

ANOTHER sophisticated way of speeding up the operation of the Electron and offering vastly increased backup memory at the same time has come from Advanced Computer Products.

The Advanced QuarterMeg Ram Cartridge, when used with ACP's DFS, can operate as sideways ram.

"It goes like a bat out of hell", says John Huddlestone, boss of ACP (0276 76545).

Amazing maze solutions

MANY eons ago I got my regal hands on an adventure which heightened my perception of text adventuring. It was released by the now defunct Magus Software and its title may ring a bell in some ears: Village of Lost Souls.

It was the most sophisticated and atmospheric text adventure I had ever seen. My only regret was that it was a BBC-only offering, and although an Electron version was promised, it never materialised.

But now the master of silicon suspense, Robert O'Leary, has bought the rights to Village of Lost Souls, and it is released this month under Robico's own label for the Electron.

It is set in a mediaeval world in which magic exists but it may only be used by those with a touch of the Talent. All use of magic is controlled by the Church, represented by the Order of Saint Leofric, Patron of Magic and discoverer of the Thirteen realms of the Arcane.

As Nathan, Inquisitor to the Order of Saint Leofric, you have been summoned to the study of Father-Magister Alain to begin an investigation for the Council of Twelve, who hold the keys to the twelve realms of order.

You are taken to the Village of Dinham where the Rector of the Church has asked for the help of an inquisitor. He believes the Lord-Talent of the Village is attempting to open a portal into the forbidden thirteenth realm, the Realm of Chaos.

Confused? You won't be, after you have first played Village of Lost Souls. This is an essential purchase for any serious adventurer. If

you think that I went over the top in my review of Lost Crystal, just wait for the review of this masterpiece.

To balance the scales, Robico has also announced that Blazing Star will not now be released, until the autumn.

I have said in the past that the success of this column is dependent upon you, the reader. That becomes evident with the wealth of help, requests and communications which I receive. This month I must thank a number of readers.

You will notice that in Hall of Fame, I have begun serialising The Lost Crystal solution, kindly sent in by Robert Henderson.

I usually receive at least one letter a month from Robert who has proved invaluable with many aspects of adventuring. In recognition of this service, I hereby bestow upon him the honour of Baron of Electron Adventures.

One recent offering from Baron Robert is the following program which will list the locations and objects in Kayleth — and, I imagine, most other recent Adventure Soft releases:

```
*LOAD KAYLETH

NEW

10FOR F=&F00 TO &6800
20IF ?F>31 AND ?F<127 VDU?F
30NEXT

RUN
```

The pupils of Mrs Nash's class of Merton Court School in Kent deserve a special vote of thanks for the wonderful drawings which they have sent. These are now adorning the many walls of my castle.

Sincere thanks too to Bob Harvey for his marvellous solution to Hampstead, which is both concise and easy to follow. Can you, Bob, or any one else provide me with a comprehensive solution to Dodgy Geezers?

I reserve special thanks for Louise Hand, age 12, of Warrington, Cheshire, who wrote me a beautiful six-sided letter. I will answer your questions in the Problems Solved section, Louise.
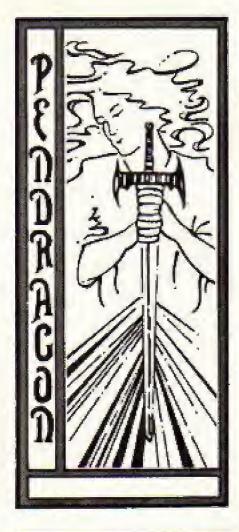
Mention should also go to Paul Chapman who took three days without food — not recommended — to complete Thunderstruck, and has sent me a full solution. I'm sorry, but I only deal with adventures, Paul.

I have been inundated with readers' own text adventures which have been entered in the competition, which I announced in the June edition of my column. The general standard of writing and breadth of imagination is staggering, and some of the puzzles have gone beyond the realms of exorbitation. Keep it up, and don't forget that the closing date for entries is July 31st.

A few months ago I promised to tell you which was my favourite commercially produced text adventure. At the time it was a toss-up between Lost Crystal and Myorem, but the calibre of recent releases has changed all that.

The following adventures are, I believe, the cream of those available for the Electron: Enthar Seven (Disc only), Village of Lost Souls, The Saga of a Spy, The Lost Crystal and Woodbury End.

That's it for this month, so, until software means comfortable clothing, happy adventuring.

OVERTURE AND BEGINNERS

This month I wish to look at some of the more devious mazes which occur in many text adventures.

These are the types of maze, where for one reason or another, it is not possible to simply drop an object in order to map your progress.

In Myorem, for example, once you have mastered your escape from the drainage ditch, you find that much of the ensuing scenario is either confusing forest or labyrinthine jungle.

If you try dropping anything in the jungle, it is soon stolen by a kleptomaniac howler monkey, making the mapping technique mentioned last month, useless.

I struggled for hours at this stage of this particular game, until it suddenly dawned on me that the elaborate but identical location descriptions had obscured my desire to SEARCH.

SEARCHing at location one in the Jungle (shown in Figure I) highlighted a broken branch to the North East.

Progressing in that direction and continuing to SEARCH, I discovered, disturbed undergrowth to the
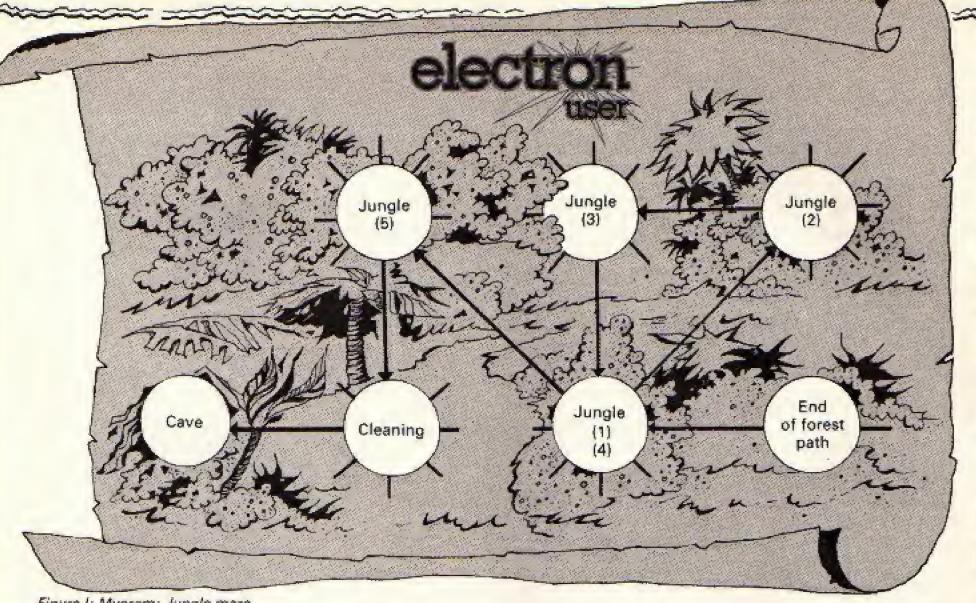
Figure I: Myorem: Jungle maze

West. Going west and investigating, unearthed grass which had been parted to the South, and so on.

In a matter of minutes I had ploughed through the jungle to an even more devious puzzle at the entrance to a cave – but that's for another day.

Robert O'Leary uses a similar type of maze in sector one of Enthar Seven, but here the emphasis is on listening rather than looking. Once again your success is only rewarded by a more perplexing problem.

Later in Myorem you will come across a series of mountain paths which are almost impossible to map by conventional means.

I only found my way through this network and avoided the talons of a ravenous condor by acquiring a map at a much earlier location. It is wise to look for quirks such as this in some of the more ingenious adventures.

In some other adventures the failure to correctly map a maze can be disastrous. In Kingdom of Hamil, I was devoured by a tyranosaurus rex for failing to keep to a logical route.

In Philosopher's Quest I

died if I did not pay attention to the wind direction in the whale's stomach.

It is often important, even desirable, to get killed a number of times in order to successfully map a maze. It does of course, go without saying that you should always SAVE your position to a blank tape before venturing into anything which may appear dangerous. You can then return to this position each time you are killed.

I remember using the explore and die technique quite extensively in earlier games such as Sadim Castle and Greedy Dwarf.

In both those adventures the annihilation factor is quite high. In fact in Greedy Dwarf, it is imperative that you own a compass to map the muddy maze and constantly stay one step ahead of Arfa in the maze.

Don't forget that with patience and careful mapping, no maze is impassable. If, however, at the end of the day you decide that mazes really are a bane which you can't tolerate, I suggest you try an Epic or Larsoft release which avoid those labyrinths like the plague.

HALL OF FAME

Enthar Seven – The Boss (continued)

You have now entered Sector 1 and exploration of this area is vital if you want to make further progress.

Pick your way along the canyon, path and ledge until you arrive at the Cavern entrance. You must now SWITCH ON your LIGHT source and continue NORTH into a short tunnel.

A quick sortie WEST will reveal a pair of boots which should be TAKEN. However, further escapades NORTH will leave you at the mercy of some bloodthirsty bats. There is at present, no way past these carnivorous creatures.

A trip NORTH EAST from the cavern will eventually lead you into a twisty maze which is easily mapped by LISTENing. This maze only leads to a more logical set of passages where lurks a little man who speaks gobbledegook.

No matter how much further you explore Sector 1 you will need to look elsewhere to solve the problem of the bats and the little man.

I recommend at this point that you make your way to the Teleport chamber – WEST of the cavern – and return to the Command Centre. Once back, it is a good idea to store all possessions here, before venturing forth again. You can always return to pick up any useful items.

Sector 4 is the zone which will furnish you with the much needed equipment and help.

Rick Hanson – Robert Hales (continued)

Leave the Butchers, go S, W, W, S, E, E, S to the dark alley. Go W, W, W, W, S and IN to the tavern. INSERT IMPI IN TO THE BANDIT then PLAY BANDIT. Get the QUHUT, go W and PAY BARTENDER WITH QUHUT. He will give you a bottle of ale. Try drinking it.

Go U, U, U, N, N, N, N and OPEN EAST DOOR then GO E into Bedroom Four. CLOSE WEST DOOR then SLEEP. You will receive further instructions in the form of a code.

Translated it reads: VISIT THE CHURCH THEN GO TO THE BOOKSHOP AND ASK FOR THE TWELVE

APOSTLES. Leave your room, go N, OPEN WEST DOOR and go W into the Bathroom. SHAVE WITH THE RAZOR.

Leave the tavern, go back to the village square. Go W, W, N, N, NE, SE, S, W and IN to the Bookshop. ASK FOR THE TWELVE APOSTLES. You will be given a book. Read it. The coded instructions read: BEYOND THE FARM-HOUSE LIES YOUR DESTINY. MEET A FELLOW AGENT AT THE ICE SHRINE.

Leave the Bookshop, go N, NW, N, NE to the Petrol Station forecourt. FILL BOTTLE WITH PETROL, then go NW, N, N, NE, E, E, E, E, S, S, W, W, S. Get the tyre, then go IN to the car and GET THE COVER.

Go N, E, E, N, N, W, W, W, W, SW, S and then W through the hole in the hedge. Go N in the muddy field to where the river is calm and SWIM.

The Lost Crystal – Robert Henderson

Part one of the game only involves a few problems, with different ways of being solved.

Your quest begins in a small field and in the distance you can see a windmill. Go to the mill, TAKE COIN and READ NOTE. Now KNOCK on the door and offer to HELP the miller with his sacks.

GO INside the mill and UP the stairs. The miller is by the sack hoist and will tell you what to do. If you wish, you may ASK him FOR the KEY. Go back down to beyond the Stone Room and OPEN TRAPDOOR.

Now you must drop all your possessions and TAKE the SACK. TIE SACK TO THE ROPE and TAKE EVERYTHING. Soon the milkman will knock, so you will need to answer the door. The miller will fall through a hole in the floor – no need to worry, he is okay.

There are two ways to get into the gallery and get the sail pole. The first is to DRINK the MILK – but keep the empty bottle – and go upstairs and open the jammed door.

The second way is to use the key which the miller may have given you, to open the mill store. There you will find a ladder which, if propped against the mill wall, will let you CLIMB UP and get the pole.

To finish part one you should GET the LAMP and climb down the rope in the mill. The rope will snap and you will have entered Part 2 of the adventure.

ELECTRON owners may be feeling a little down in the dumps after reading the superb report on the new Acorn Archimedes computer in the August issue of *The Micro User*.

A minimum of half a megabyte of memory, an incredibly fast 32 bit RISC chip and superb graphics make it a fantastic micro, and very desirable.

One of the demonstrations that the *Micro User* team saw running on the Archimedes when they visited Acorn consisted of a large animated ball bouncing round the monitor screen. Behind it was drawn a grid pattern.

The programming involved isn't as easy as you might at first think.

To print the ball in front of the grid you must first store the section of screen memory where the ball is to be placed. Then a plain mask is printed over this part of the grid and finally the ball is printed on top of the mask.

To move the ball, it is erased by replacing the stored screen memory and the whole printing process is started again. This is quite

Bouncing with the best!
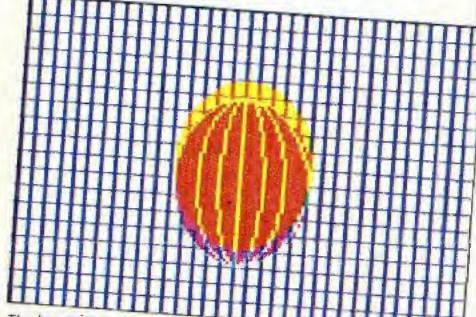
JOANNE STEVENS brings some of the power of the Archimedes to the Electrom

a well known demonstration and was first seen on the Commodore Amiga.

It's used to demonstrate the power, speed, memory and flexibility of a technically advanced computer, implying that lesser micros aren't capable of such things.

This simply isn't true. The program presented here is an extremely fast bouncing ball demonstration for the Electron. In fact, a short delay is built in to slow it down!

Turbo owners will find the animation slightly smoother when it is switched on. However, it won't work in 64k shadow ram mode as the screen memory is

The bouncing ball demonstration

accessed directly.

Now whenever you hear an Archimedes owner boasting about the power of his micro, show him what your Electron can do!

```
   10 REM Bouncing Ball
   20 REM By Joanne Stevens
   30 REM (c) Electron User
   40 DIM ball% 2000
   50 ENVELOPE 1,131,1,1,1,1
0,10,10,126,0,0,-126,126,126
   60 MODE 5:*FX16
   70 VDU23,1,0;0;0;0;
   80 VDU 19,2,4;0;19,3,3;0;
   90
  100 REM Grid
  110 GCOL 0,2
  120 FOR x=0 TO 1248 STEP32
  130 MOVE x,0:DRAW x,29*32
  140 NEXT
  150 FORy=0 TO 29*32 STEP32
  160 MOVE 0,y:DRAW 1248,y
  170 NEXT
  180
  190 REM Ball
  200 GCOL 0,1
  210 R%=158
  220 VDU 29,6*32;23*32;
  230 FOR angle=0 TO 340 STE
P 20
  240 MOVE 0,0
  250 MOVE R%*COS(RADangle),
R%*SIN(RADangle)
  260 PLOT 85,R%*COS(RAD(ang
le+20)),R%*SIN(RAD(angle+20)
)
  270 NEXT
  280 GCOL 0,3
  290 FOR i=-R% TO R% STEP R
%/3
  300 MOVE 0,R%*SIN(RAD90)
  310 FOR angle=90 TO 270 ST
EP 20
  320 DRAW i*COS(RADangle),R
%*SIN(RADangle)
  330 NEXT
  340 NEXT
  350
  360 REM Store Ball
  370 PRINT TAB(5,1)"Thinkin
g..."
  380 A%=&5800+3*&140:B%=bal
l%
  390 FOR i=0 TO 11
  400 FOR j=0 TO 12*8-1
  410 ?B%=?(A%+i*&140+j)
  420 B%=B%+1
  430 NEXT
  440 NEXT
  450 PRINT TAB(2,1)"A Meagr
e Demo..."
  460
  470 REM Machine Code
  480 x=&70:y=&71
  490 xmax=26:ymax=17
  500 xdir=&72:ydir=&73
  510 addr=&74
  520 rows=&7A
  530 FOR pass=0 TO 2 STEP 2
  540 P%=&900
  550 [OPT pass
  560 .start
  570 LDA #A% MOD256:STA add
r:LDA #A% DIV256:STA addr+1
  580 LDA #1:STA xdir:STA yd
ir \left+down
  590 LDA #xmax+1:STA x:LDA
#ymax+1:STA y
  600 .loop
  610 LDA #19:JSR &FFF4
  620 LDA #19:JSR &FFF4
  630 DEC x:BNE notx
  640 LDA #7:LDX #sound MOD2
56:LDY #sound DIV256:JSR &FF
F1
  650 LDA #xmax:STA x
  660 LDA xdir:EOR #1:STA xd
ir
  670 .notx
  680 LDA xdir:BNE right
  690 SEC:LDA addr:SBC #8:ST
A addr:LDA addr+1:SBC #0:STA
 addr+1
  700 JMP updown
  710 .right
  720 CLC:LDA addr:ADC #8:ST
A addr:LDA addr+1:ADC #0:STA
 addr+1
  730 .updown
  740 DEC y:BNE noty
  750 LDA #7:LDX #sound MOD2
56:LDY #sound DIV256:JSR &FF
F1
  760 LDA #ymax:STA y
  770 LDA ydir:EOR #1:STA yd
ir
  780 .noty
  790 LDA ydir:BNE down
  800 SEC:LDA addr:SBC #&40:
STA addr:LDA addr+1:SBC #1:S
TA addr+1
  810 JMP print
  820 .down
  830 CLC:LDA addr:ADC #&40:
STA addr:LDA addr+1:ADC #1:S
TA addr+1
  840
  850 .print
  860 LDA #ball% MOD256:STA
data+1:LDA #ball% DIV256:STA
 data+2
  870 LDA addr:STA temp+1:LD
A addr+1:STA temp+2
  880 LDX #11
  890 CLC
  900 SEI:LDA #4:.fx19 BIT &
FE00:BEQ fx19
  910 .loop2
  920 LDY #12*8-1
  930 .loop3
  940 .data LDA &8000,Y
  950 .temp STA &8000,Y
  960 DEY:BPL loop3
  970 LDA temp+1:ADC #&40:ST
A temp+1:LDA temp+2:ADC #1:S
TA temp+2
  980 LDA data+1:ADC #12*8:S
TA data+1:LDA data+2:ADC #0:
STA data+2
  990 DEX:BPL loop2
 1000 CLI
 1010 LDA #129:LDX #256-113:
LDY #&FF:JSR &FFF4
 1020 TYA:BNE exit
 1030 JMP loop
 1040 .exit
 1050 RTS
 1060
 1070 .sound
 1080 EQUW &11
 1090 EQUW 1
 1100 EQUW 0
 1110 EQUW 5
 1120 ]
 1130 NEXT
 1140 CALL start
```

ROMBOX PLUS

SLOGGER's answer to the PLUS 1

Q. Why not buy the Acorn Plus 1?
A. Because "Excellent though the Plus 1 is, I think there is still room for improvement" said the Electron User Magazine.

The ROMBOX PLUS has this improved specification:
1. Four ROM/RAM sockets to free the cartridge slots for more important add-ons. Each socket can take the usual 8K/16K ROMs but can also take 16K RAMS offering a staggering 64K SIDEWAYS RAM.
2. The PLUS1 Cartridge slots.
3. The PLUS1 centronics printer port.
4. The Expansion ROM 2.0
5. Switched Joystick Interface available.
6. Analogue Joystick Interface under development.

Still Only £54.95

ACORN PLUS1s

Slogger has on offer a limited number of Acorn PLUS1s. These units are brand new but nonetheless fully tested in Sloggers factory.

ONLY £44.95

(£50.00 WITH EXP ROM 2.0)

TAPE TO DISK ROMS VERSION 2

* Transfer the majority of Tape programs to disk
* Acornsoft, Superior and Micropower no problem
* Compatible with programs copied using Version 1
* More successful than ever before

T2P3 for the Acorn Plus 3
T2CU for the Cumana DFS
T2P4 for AP4 and EOO DFS
T2SD for the Solidisk DFS
T2SEDFS for the SEDFS

ONLY £24.95

* Upgrade 1-2 only £5 with original ROM

PRINTER

AMSTRAD DMP 2000

* Full EPSON compatibility
* Ideal for listings, Word Processing and graphics
* Prints 105 characters per second in draft mode
* 52 characters per second in Near Letter Quality (NLQ)
* Built in 2k buffer to increase throughput
* FREE lead suitable for ROMBOX PLUS or PLUS 1

Excellent value at only £185

JOYSTICK INTERFACE

for ROMBOX PLUS or ACORN PLUS 1
* Uses cheap ATARI–type joysticks
* Compatible with ALL Tape and Disk systems.
* Emulation of keys by Expansion ROM 2.0
* Can be used with well written games either
   WITH OR WITHOUT A JOYSTICK OPTION
      "The ultimate Joystick Interface"
                                   A&B July 87
   "Well worth saving up for, I can recommend it to all Arcade Addicts"
                          Electron User May 87
      Only £14.95
* Requires Electron Expansion 2.0
   Only £5 with this unit

JOYSTICKS

Switched Joysticks for the Slogger Joystick Interface
Quickshot I   £4.00
Quickshot II   £6.00

SLOGGER EPROM CARTRIDGE

* Carries TWO 8K/16K ROM/EPROM software
* Compatible with PLUS 1, ROMBOX PLUS and BBC Master.
* Simply plugs into cartridge slot
      £10.00 for a limited period

ROM SOFTWARE

STARWORD	£34.50
STARSTORE 2	£29.95
SEDFS upgrade ROM for CUMANA owners	£24.95

* BARGAIN CORNER *

STARSTORE	£9.95
STARMON	£9.95
STARGRAPH	£9.95
PRINTER ROM	£9.95
STAR TREK	£9.95
ELKMAN	£9.95
PDG	£9.95

OTHER PRODUCTS

5.25" 40/80 Track D/S disk drive inc. PSU	£149.00
3.5" 40/80 Track D/S disk drive inc. PSU	£149.00
Printer Lead (for ROMBOX PLUS/PLUS 1)	£9.95
Electron Advanced User Guide	£3.95
Vine Micros ADDCOMM	£26.00
Vine Micros MATRIX ROM	£36.00
USER PORT for any DIGITAL MOUSE	£29.95

32K SIDEWAYS RAM CARTRIDGE

for ROMBOX PLUS or PLUS 1

* Ever popular on the BBC
* Best used with EXP ROM 2.0
(1) Save ROM backups for loading into Sideways Ram.
(2) Set up a 16k Print Buffer
* Splits as 2x16k pages
   VIEW and VIEWSHEET in one slot!!!
* Write protect option . . . prevents corruption of software.
      "A very powerful piece of hardware"
                          Electron User July 87
ONLY £34.50
(£39.50 with EXP ROM 2.0)

EXPANSION ROM 2.0

for ROMBOX PLUS or PLUS 1

* Supports Printer Buffer with 8K/16K sideways RAM or 12K with MASTER RAM BOARD.
* RSAVE command to save ROM images to tape or disc.
* RLOAD command to load Sideways RAM with a ROM image.
* ROMS command to displays ROM/RAM in system.
* JOYSTICK command for the SLOGGER Joystick Interface
* Allows cassette loading in high resolution.
* Supports all standard functions (Printer, ADC, RS423)
* Simply fits into internal ROM socket.
   "There is room for improvement in the operating system. No doubt this was the idea behind Expansion 2"
                          Electron User, July 87
Only £11.95
(£5 if purchased with 32K SIDEWAYS RAM or JOYSTICK INTERFACE)

SLOGGER'S AUTHORISED DEALERS

Aberdeen Micro Logic	0569 86233
Birmingham Allen James Computers	021 778 5757
Bristol Avon Computer Exchange	0272 537981
Bromley Datastore	01 460 8991
Burnley Atomic Computer Systems	0282 54299
Byfleet Calancraft	0932 342137
Cardiff Computer Exchange	0222 483069
Cardigan Cardigan Electronics	0239 514483
Chesham Reeds Photo & Computing	0494 783373
Croydon Computer Centre	01 683 2646
Fareham Project Expansions	0329 221109
Farnham Farnham Computers	0252 723107
Glasgow Computer Depot	041 332 3944
Leicester DA Computers	0533 549407
London Compuface	01 580 0151
Luton Hobbyte	0582 457195
Market Harborough Harborough Computers	0858 635280
Newbury M&S Associates	0635 45774
Sheemess Swaletek Systems	0795 665530
Stockport Dram Electronics	061 429 0626
Telford A1 Computer Supplies	0952 502737
Trowbridge West Wilts Micros	02214 62759
Ulverston Skand Computers	0229 56984
Warrington Cheshire Micros	0925 414109
Netherlands Velobyte Computers Rotterdam	010 4138197
Mail Order Only	
Milton Keynes Towerhill	0908 74246
Stockport 21st Software	0625 528385

All prices include VAT P&P UK Mainland only

Boring battling

THIS package contains, Mountains of Ket, Temple of Vran and The Final Mission — which form the classic and colossal Ket Trilogy.

Often when a piece of software is given enough sales hype to sell sand to the Saudis you becomes a bit sceptical about the quality of the game.

The adventure is hardly colossal. The Mountains of Ket only has about 66 locations, and the complete trilogy has no more rooms than an average Epic or Robico release.

To its credit, it covers three tightly linked Sphinx/Ring of Time type adventures which contain some excellent chaining puzzles and devious passwords.
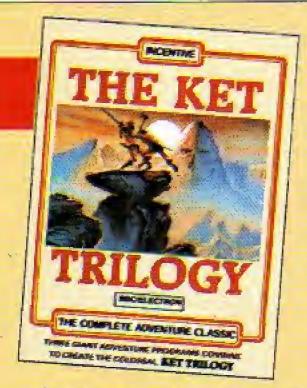
Unfortunately things begin to break down in the dungeons and dragons style combats which seem to occur at every sixth location.

These combats are supposedly based on the prowess, energy and luck factors of you and your foes, yet are nothing more than a rather tedious series of random number generations.

Conflict can be avoided in some cases by trading wares or, in the case of the Ogre, by a spot of illicit gambling. However, your whole progress can be ruined by unaccounted probability.

The room descriptions are sparse and little atmosphere is created. When any does exist, it is destroyed by silly interjections such as when the password to a secret door in mint condition is Polo. That type of humour may be at home in Terrormolinos but is out of place here.

The whole approach seems rather dated, with the parser and text compression being extremely limited.

It is a shame that the memory taken up by the combats and fancy screen display couldn't be better utilised by creating more locations.

I'm not too keen on this adventure, as there are better examples.

Pendragon

Presentation	8
Atmosphere	3
Frustration Factor	7
Value for Money	3
Overall	5

Superb adventure

THE Lost Crystal has been more than two years in the making and supposed release dates have come and gone. However, the wait was worth it. This is one of the best text adventures I have ever seen.

It is offered in a superlative manner, with two cassettes packaged with help and instruction booklets in a scintillating presentation box. The screen shots on the insert give an indication of the marvellous graphics ahead.

The documentation helps set the scene for the 400 location magnum-opus which follows. "The Rainbow Crystal which has protected the people of Zaloria for many years," (from the hideous Bogle and his marauding Morgs), "has been struck by a mysterious bolt of lightning and shattered into seven pieces".

You have been chosen to recover the seven crystals and find a way to recreate the Rainbow Crystal and so restore peace to the Freelands.

You begin this massive jaunt standing to the south of a large windmill. An exploration of the mill and its storehouse should let you solve these early puzzles which are really just beginners' fare. Though you may get a shock when the milkman knocks for his money. Don't forget that "Milk has gotta lotta bottle."

In part two, the going gets tough and I had to resort on at least one occasion, to the excellent help book.

Careful progress through the many twisty passages (don't worry it isn't a maze) and assistance from the strong but cowardly Miller should help you to overcome these hurdles. But, don't waste that coin.

The Bogle is soon encountered and the poor Miller must be scarificed to save your own skin. Who said Epic was heathen.

The third part is the largest and most difficult and involves cunning manipulation of objects, some excellent chaining puzzles and the start of your collection of lost crystals.

You will now discover the use for that sail pole you have been carrying since the beginning of the game.

A secret passage will reveal a princess who is really a trading post for some of the possessions you are carrying.

In part four, you become involved in a titanic struggle to rescue Merlin. You must defeat the hydra, a dragon and a vampire before being reunited with an old acquaintance. The end is both spectacular and amusing.

My only criticism of the game is that the colours are sometimes a little garish.

The puzzles are, on the whole, logical and typical of the Epic style of construction.

The most important character in the game — besides yourself — is the Miller, who demands sympathy and affection. I adored his comments at various stages of the game and his muscular help is required on more than one occasion.

This is a rare adventure which has a lot to offer the beginner and expert. A whole new experience which must be savoured and an adventure which I whole heartedly recommend.

Pendragon

Presentation	10
Atmosphere	10
Frustration Factor	9
Value for Money	10
Overall	10

Race against time

I HAVE to admit I must be one of the few people that didn't like Repton or its many sequels. On the face of it, Survivors seems like Repton with a different scenario, though the original idea is a bit more sinister.

It all happens in the year 2087 after a massive nuclear war has devastated Earth. Your job is to find and rescue the cryogenically suspended survivors of the holocaust from the crumbling remains of a hibernation dome.

Three specially armoured rescue droids have been sent into the area to help and are now under your control.

Each droid has its own special abilities. The first is a high speed tunnelling machine, the second a mobile teleporter and the third a bulldozer droid for shifting large rockfalls in the area. But it is not going to be easy.

From the moment the title page weaves on to the screen you begin to get the feeling that you are in for a mindbending task. Press S to start the game (after all who needs instructions?) and you're ready to go.

The screen display shows the power levels of the three robots, the score, time remaining, people left and most important, people saved.

Actually saving someone is a doddle, all you do is move over them with droid number two, the teleporter, and they're whisked to safety.

The problems lie in finding where they are, and then rescuing them without getting trapped yourself.

Now that is not quite as easy as you might think and your route has to be very carefully thought out. One wrong move can bring boulders bouncing down, blocking the escape route.

Survivors becomes more of a challenge when you realise that the three droids have to be used intelligently.

Initially you can use the tunnelling machine to dig a path through the rubble and then use the teleporter to pick up survivors. But you will soon have to start shoving rocks around, and then it starts to get difficult.

You have to cope with the old maintenance droids as well. These once peaceful machines, having been damaged by radiation, are now on the loose and pose a severe threat to any of your own droids that touch them.

Unlike most games of this ilk, touching the enemy does not result in instant death, merely the loss of power. If the power level of any of the three droids falls to zero then you will have to start all over again.

When you do get through level one there are another six to come, and judging by the challenge of the first it may take a long time.

The game is addictive and aided by well-animated and colourful characters. One minor criticism is the sound which is rather a disappointment, the only regular noise being the relentless ticking of the clock, counting your remaining time.

I would have liked to see the addition of a joystick option and screen designer, although this is more personal preference rather than a necessity.

All in all, at this pocket-money price, Survivors represents extremely good value for money and could well knock Repton off it's perch.

Mark Smiddy

Sound	4
Graphics	9
Playability	10
Value	10
Overall	9

Bag that birdie

BIRD Strike is a recently rereleased title from the Firebird Silver budget range.

The game starts in the first World War, with biplanes and carrier pigeons, but the idea is nevertheless simple. Move a gun emplacement from side to side along the bottom of the screen — if anything moves, shoot it.

A great deal of attention has been devoted to the background detail and the result is a simple, but surprisingly effective Belgian rural scene with cottages, trees and a country church. My only complaint is that your gun emplacement escaped this attention.

Six biplanes lurk above the cloud cover. One by one they swoop, rapidly homing in on you, showering bombs as they come. You can loose off two shots at once — you certainly need to.

A direct hit on the centre of the plane causes it to vanish in a cloud of smoke, reappearing as a small cross in the graveyard of the church. Destroy all six to move on to the next screen.

If your marksmanship is less than perfect, your quarry will disappear off the bottom of the screen and dive again. Needless to say, any contact is fatal and even a wingtip is enough to flatten you, leaving only a skull as your memorial.

If you hit a plane but only clip its wing, a delightfully animated carrier pigeon is released, cheeping merrily, wings flapping. Shooting the birdie adds a note to the music staff at the top of the screen. After 12 pigeons have been brought down the screen ends, you are awarded a substantial bonus and a tune is played.

Your attackers are now World War II single-wing fighters. You have a similarly unhealthy attraction for them, but can deal with them the same way.

After two more screens — jet fighters and then helicopters — you are once again back in 1914 with no apparent increase in difficulty.

I found the game very enjoyable — my only complaint is the way it slows down when simultaneously displaying a plane, pigeon and gun emplacement. It rather takes the edge off the fast action.

Martin Reed

Sound	7
Graphics	8
Presentation	7
Value for Money	9
Overall	8

Darn near catatonic

SOFTWARE at a reasonable price is always welcome and it's nice to see more appearing on the market. But we want reasonable value for money and I feel that this offering falls short.

Squeakaliser, a maze type game with you playing the mouse world's answer to Clint Eastwood, didn't make my day.

The half-pint-sized hero must gather a number of keys to open a box containing a giant cat-trap. This is moved round until all the cats have been eliminated from that screen.

On alternate screens the box opens to reveal sticks of dynamite that have to be handed to the cats. Unfortunately, sometimes the cat hands it back and runs off.

The idea behind the game is good. Movement of the hero is simply up, down, left and right. The sound is adequate and the graphics are very cute indeed — when the hero gets his comeuppance the little birds twittering round his head are plain to see. But the game is just not properly finished off.

The opening screen starts the rot with the title, Squeekaliser, which shows that someone, somewhere can't spell.

The game uses part of the screen for memory resulting in the picture being limited by two very distracting bands of colour at the top and bottom of the display. I cannot believe such a simple game should use that amount of memory.

One result of the limited playing area is that the remaining screen is too small to generate anything other than a very basic maze.

When the game starts you are offered a choice of two levels, known as Sheer Panic and Sweaty Palms. These would be better renamed as Sheer Boredom and Sweat it Out. The only difference between the two is that choosing the second causes all the characters to slow down to the point where it is almost impossible to cross the screen before the time runs out.

Another point of frustration is that although your mouse starts with seven lives, once he is cornered and despatched by a cat, he is reincarnated in exactly the same place. This usually means the loss of all your lives.

Oh Bug-Byte, Plan B was terrific, I liked Dunjunz so much I went out and bought it myself, but for me Squeakaliser is a CATastrophe.

Beejay

Sound	4
Graphics	3
Playability	2
Value	4
Overall	3

All abroad with Repton

SINCE the successful launch of Repton 3, Superior has been inundated with letters asking when it would release a new set of screens.

Well it is my great pleasure to announce that it's here, entitled Around the World in 40 Screens. The package consists of five files, each containing eight mind blowing screens.

The cassette represents excellent value for money since not only do you get 40 new screens, there is also a copy of Repton 3 and its own screen editor. So, if you didn't buy it first time round, here's your chance to see what you've been missing.

As the title suggests, each of the five files is based around a different country or environment. The characters all retain their original characteristics, so boulders fall, eggs hatch and monsters chase, but every last one of them has been redefined to match their new surroundings.

Repton's voyage begins in the Wild West where 'Clint' Repton has to clear the town of hamburgers inside a strict time limit.

The boulders have been replaced by wagon wheels, the monsters are now Red Indians and the spirits have changed into bandits who must be jailed.

The remaining four files are set in the snows of the Arctic, the mysterious Orient, the ocean depths and Africa. A great deal of time and effort has gone into creating a graphical treat, and a series of puzzles that are just as testing as the first collection.

In the original Repton 3 you had the chance to load a new series of screens if you grew bored with the first set, but you were still playing with the same old characters.

By using a brand new character set with each group of screens your attention is held much longer — it's now far easier to keep the eyelids open at three in the morning.

Unfortunately, the vastly different character sets cause a few problems when loading a new file. You don't know which are safe to touch and invariably lose a few lives.

The best course is to use your first venture into a new landscape as a suicide exercise, walking blindly into the different characters and noting what happens.

The inherent expandability of Repton 3 has been exploited to the full by Superior. If it continues to release extensions such as this, our little green skinned buddy could be around for many a year.

Jon Revis

Sound	6
Graphics	8
Playability	10
Value	10
Overall	9

Touch of luxury

MARK SMIDDY reviews ACP's Advanced Control Panel

MANY years ago TV and monitor screens were not used to display output from the early computers. Instead, they relied upon a panel of indicator lights.

Similar to those seen in many science fiction films from the fifties, they were known as front-end control panels, and became extinct with the advent of the monitor.

Now ACP has revived the control panel idea, giving Electron users a friendly front-end panel.

Replacing the bulky electronic display, it is supplied as software on rom. To use it you will need a Plus 1 with a suitable rom cartridge or Rombox.

The panel appears instantly on power-up, taking control of the machine instead of Basic. At this point the only major problem many Electron users are likely to encounter becomes apparent — the display is unalterably in Mode 0.

This is necessary for the software but is a nuisance if your screen can't display 80 column text clearly — and this means the majority of televisions.

The control panel provides you with a very convenient interface between you and the micro's more complex functions.

The intitial control panel display has four window headers — Language, Mos, File and Panel.

Highlighting Language,

using the cursor keys, brings up a language window.

Under this heading are listed all roms present in the machine that announce themselves as languages. Normally this will be just Advanced Control Panel itself, Basic and the Plus 1 rom.

It is important to note that language roms are not necessarily programming languages, just that their designers included a language entry point in the software. The Plus 1 rom can't be selected even though it appears on the list.

If you have the language cartridges View, Viewsheet, Lisp and so on, these will appear on the menu. Entering a rom from the panel is a simple matter of highlighting the one you wish to use and pressing Return.

Under the heading Mos lie the operating system functions. Some functions listed under this menu and further sub-menus do not directly concern the Electron — the rom can be used on the BBC

Figure II: The Mos menu

Micro and Master as well.

Clock, for instance, accesses the real time clock in the Master. Trying to access one of these extra functions on the Electron results in the error message: "Not supported".

This should not be a problem, and if you ever upgrade to the heady heights of the BBC Master, you'll have one less rom to replace.

Available under the same heading is a pop-up-calculator that allows conversion between decimal, binary and hexadecimal and simple arithmetic to be performed.

It's a lot easier to use this than to perform the same feats from Basic.

Another useful function under the same heading is the rom list which lists the 16 rom locations allowed by the operating system and their contents, if any.

In this window it is possible to switch off (in effect unplug) any of the roms.

This can be useful for stopping one rom accepting a star command intended for another.

Under the File menu are various functions concerned with filing systems, information on files, changing drive with disc systems and so on.

Finally, most powerful of all, is the Panel menu itself. This enables you to tailor the panel or design a completely new one.

It is possible, for instance, to create a panel which only contains those functions directly relevant to the Electron. Although this is not strictly necessary, it shows what can be achieved.

User defined panels can be saved to the current filing system, ADFS disc for example, and then the panel can be re-entered at a later date with *ACP followed by the filename.

Advanced Control Panel is supplied with a comprehensive 29 page manual that contains everything to get you started from fitting the rom right up to the more advanced features of the software.

It is a reasonably user-friendly piece of software, and although a luxury item I can recommend it to anyone who wants easy access to the Electron's functions.

One word of warning before you rush out and buy it — make certain that you can easily read 80 column text on your television or monitor. ∎
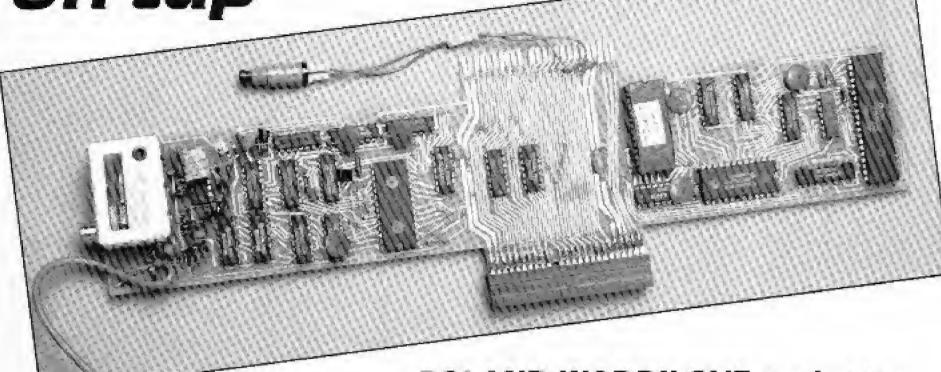
Figure I: The disc catalogue under ACP control

Hardware Review

Teletext on tap

**ROLAND WADDILOVE evaluates
a Mode 7 adapter for the Electron**

IT has taken nearly four years, but now here at last is a true Mode 7 adapter for the Electron. And very impressive it is too.

It's a hardware add-on which plugs directly into the back of the Electron and is about the same size as a Plus 1.

The pre-production prototype version I tried was not cased, so I can't say what it will look like when finished. Hopefully, it will match the colour and style of the Electron.

Plus 1, 3 and Rombox owners needn't worry, as the edge connector is continued at the rear of the board and our Rombox Plus and Cumana disc interface worked perfectly throughout the review.

There is a short monitor lead on the left side of the board which is not, as I first thought, for plugging into a monitor, but into the monitor socket on the Electron's side.

The TV output is taken from the Mode 7 adapter itself, not the Electron. There isn't a monitor output, and I hope this slight deficiency is rectified on production models.

There's really very little to it. You simply plug in, switch on, type *MODE7ON and tap the Break key. You now have Mode 7 in addition to the normal modes 0 to 6.

HIMEM is set at &7C00 so 5k extra ram is available for your programs.

The Electron has a habit of clearing this on pressing Break (it still thinks this is the screen memory) so there's a special reset button which acts like a soft Break. However, the content of the extra ram stays intact.

You can print all teletext characters, colours and graphics on the Mode 7 screen and you can even poke it directly if you wish.

As a test I borrowed half a dozen BBC discs from the *Micro User* team and booted them up on the Electron. They all worked.

In fact, no matter how they were written – legally or illegally – they produced a perfect display every time.

One of the toughest tests was Invasion from the February 1984 issue of *The Micro User*. This is a Mode 7 version of space invaders.

After adding two lines to stop the introductory music from playing it ran first time. It was every bit as good on the Electron as it is on the BBC Micro.

As a bonus, the adapter also works with Slogger's Turbo (but not 64k shadow ram mode), so now you can have the speed of the BBC Micro and Mode 7 as well.

With this combination quite a high proportion of (unprotected) BBC Micro software will run on the Electron.

But you won't be able to run commercial software such as Acornsoft's Revs.

There are many reasons why this won't work. One is simply that the software checks which micro it is running on while loading and will stop if it's an Electron.

You can turn the adapter off at any time so the micro behaves as a normal Electron.

Several new commands have been added to the Electron's operating system. The Mode 7 display can be brought down the screen with *TV255 and the BBC Micro's red function keys are emulated on the Electron's keyboard.

On the BBC Micro you can press Shift, Control or Shift+Control and a function key to obtain special effects.

This doesn't work on the Electron, but after *EFN and Break the bottom three rows of the keyboard emulate these keys when used with Caps Lk/Func.

At £79 the adapter costs as much as an Electron itself and must be considered a luxury rather than a necessity.

Remember, no matter what you add to an Electron, it will never be exactly the same as a BBC Micro, and you could end up paying out more.

However it does carry the Electron a long way down the road towards that great micro, and if money is no object then I can recommend it. ∎

. . . up to £5 on nine quality educational programs

Full-length programs designed to stimulate, educate and entertain – in school and at home

Windmill: *Word, number and colour recognition*

Angler: *Fun with angle estimation*

Spelldroid: *Learn to spell with our friendly robot*

Educational Computing on the Electron
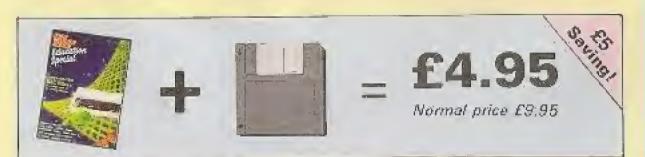
Volume 2 of *The Micro User* Education Special contains nine full length programs written to the highest standards and each picked to combine educational worth with sheer enjoyment. The nine programs cover topics from early reading and simple sums to the rules of punctuation and angle estimation – and there's an excellent introductory database.

The programs on the tape and disc have been adapted for the Electron and the magazine contains all the original listings together with advice on how they can be adapted to cater for individual needs.

Covers all age ranges from infants to secondary.

Tortal: *Teach and test the rules of single addition*

PLUS:
Database: *A comprehensive database for the young learner.*
Chinese Takeaway: *Teach and test the rules of simple subtraction.*
Fun Factors: *Arcade style factor learning.*
Discovery: *A strategy based phrase identification game.*
Punctuation: *Test and teach the rules of punctuation.*

£2 Saving!

+ = **£3.95**
Normal price £5.95

£5 Saving!

+ = **£4.95**
Normal price £9.95
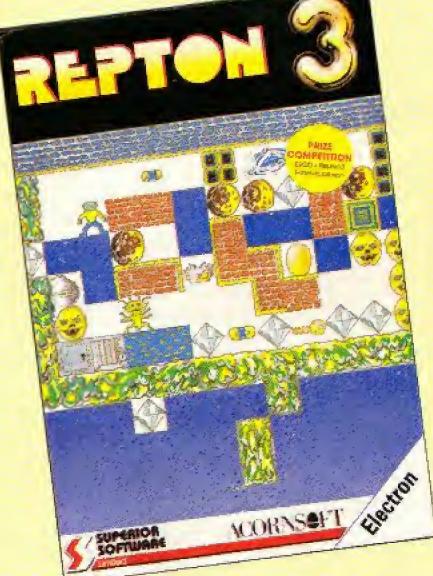
Devise your own Repton screens ... and baffle your friends!

First there was Repton, voted Game of the Year in 1985. Then came the universally acclaimed Repton 2 with a whole set of new challenges. Now comes Repton 3 – all the fun of the first two games but with even more appeal!

Not only are there 24 screens with a host of brand new challenges, there is also a feature-packed screen designer.

Now you can devise your own fiendish screens for your family and friends to solve. Position timebombs, sprinkle the growing fungus, and set devious puzzles of rocks and diamonds. Then you can save your masterpiece to cassette or disc.

It's certainly a game which will keep you entertained for months!

You can buy Repton 3 from us and **SAVE £2** on the normal selling price.

Or you can take out a subscription and **SAVE a massive £5**.

Design a screen then playtest it

You can even change the characters

SAVE UP TO £5

	RRP	Special reader offer	YOU SAVE	Offer including subscription	YOU SAVE
Cassette	£9.95	£7.95	£2	£16.95	£5

TO ORDER PLEASE USE THE FORM ON PAGE 53

Let's keep it clean . . .

MARK SMIDDY sorts out your cassette loading problems

IF YOU use tape to load and save your programs regularly then I expect that you've sometimes had the annoying problem of your cassette player refusing to load a program.

How often have you had Data? Block? and Rewind tape messages?

Fortunately this seldom happens. But when it does what can you do when things go wrong?

Blood pressure rises, nerves become frayed and temperatures reach boiling point. Don't despair, these problems can be overcome.

The record/playback head in a tape player is an extremely fragile and sensitive piece of equipment. It reads a fine track of magnetic information off the tape and passes it on to the electronics where it is amplified thousands of times.

Therefore any error which occurs on the tape is also amplified greatly.

Your Electron is very sensitive to these problems, that's why it will rudely print Data? right in the middle of a program that seems to be loading perfectly.

Loading, is in fact quite a complex operation. When a program is saved it is split up into small sections and each section or block saved is made up of two parts.

The first part or header, contains information about the file – its name, load and execution address, the block number, length and Cyclic Redundancy Check or CRC.

The second part contains the actual data and is made

up of a long string of bytes. There can be 65535, but 256 is the usual maximum. Tagged on to the end of the data is another CRC.

The CRC is a checksum calculated using every single byte of data. It is sensitive enough to detect errors of just one incorrect bit in many thousands.

When the operating system finds that the CRC on tape does not tally with the one that it has calculated, it informs you of the error and asks you to rewind the tape a little.

Holding the rewind button down for second or so is usually enough. This can occur in either the header or data section of the block hence the errors Header? and Data?

But what about Block? This occurs when for some reason the operating system loses track of the block should be loading from tape.

It keeps a count of the last block number that loaded and expects to find the next one in sequence. Block? is not a serious error and most often occurs when the header has become corrupted.

Cassette tapes are covered in a magnetically sensitive material. Ferric oxide is the most common used, but what most people don't realise is that these materials are just very fine powders glued to a backing material and there is approximately 6750 feet of tape in a C60.

You may well ask "What difference does all of this make?". The answer quite

simply boils down to the record/playback head. As the tape is dragged over it at a rate of almost 1.9 inches per second a certain amount of friction occurs.

This causes the tape to shed a very small amount of oxide over the head, capstan and pinch roller. Figure I shows the positions of these items.

Over a period this deposit builds up to the point at

which it starts to cause problems. Woolly sound, tape skid (wow and flutter), or magnetism.

These problems can be identified by listening to a music cassette that you have recorded yourself on a good hi-fi cassette recorder.

Dirt on the head or magnetism causes the music to sound woolly or dull and lose volume, skid is fairly obvious.

None of these problems are particularly difficult or expensive to put right if caught in time, and can usually be solved by the application of a proprietry brand of tape head cleaner and demagnetiser.

The problem with cheap, poor quality cassettes is that they shed their oxide at a much higher rate and

therefore problems occur much faster.

Just to add insult to injury, this dust is highly abrasive and if you are using a high quality player then you are quite likely to do severe damage to the head. Would you polish the family silver with sandpaper?

Use cheap tapes and that's effectively what you are doing to the record/
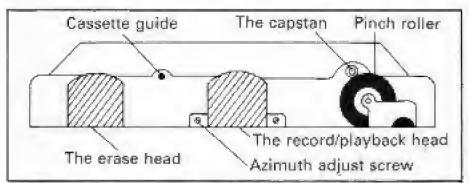
Figure I: The positions of the record and playback components.

playback head. Assuming that the head survives this battering the pinch roller invariably suffers.

In fact it is usually due to dirt on the pinch roller that causes tape decks to 'eat' tapes, leaving you with many feet of useless tape crammed inside your recorder.

The answer is to use a cleaning tape and use it often. Every eight hours of play is usually sufficient. Wet cleaning systems are always preferable since they tend to do much less damage. I have resorted to cotton buds and industrial alcohol as these are usually adequate.

If the tape is still wowing after a thorough clean then

◀ **From Page 23**

the fault is most likely to be in either the deck or the tape itself.

Again, cheap tapes are most likely to suffer from or cause this type of problem. The spools may become partially jammed inside the cassette.

This can sometimes be cured by fast forwarding and rewinding the cassette a few times, though usually a replacement of better quality is the only option.

If several cassettes seem to produce the same amount of wow, then the problem most likely lies inside the mechanism of the deck itself. This sort of trouble should always be referred to a qualified engineer.

Another problem that

sometimes crops up with cheap tapes, is tape stretch. This is why manufacturers of tape decks recommend that you avoid the use of C120 cassettes.

In both cases, the backing material used is very thin and a lot of rewinding can literally cause the tape to get longer and thinner and eventually snap. In the case of computers, tape stretch would cause data loss long before a snap occurs.

No matter how much you spend on a cassette though, never touch the surface of the tape as this may ruin that section irrepairably and corrupt any data that it holds.

That covers the tape problems, but there is another problem of azimuth angle.

This can be thought of as being the alignment of the
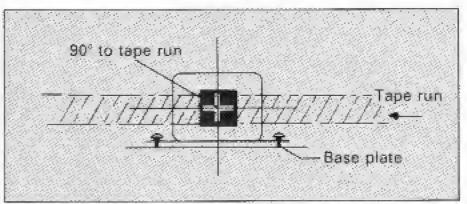
Figure II: The record and playback head showing azimuth angle.

tape head with respect to a perpendicular drawn through the tape run. As Figure II shows, 90 degrees is the angle for compact cassettes.

If after cleaning the tape heads the sound is still not crisp and clear you can try adjusting the azimuth angle. Place a cassette known to be of good quality in the recorder and press play.

Listen to the sound and

with a small screwdriver turn the adjusting screw slowly left and right until the sound is at its brightest. (Never turn the screw more than one full turn in either direction.) Do the same for the B side of the cassette. The head will now be correctly aligned.

The Electron is a reliable loader and these hints and tips should ensure that you have trouble free loading.

SO far in this short series exploring the programming language Lisp we've examined some of the simpler built-in functions such as CAR, CDR, SETQ and the predicates.

Now it's time to get down to some real programming. We'll see how to define our own functions and write our first Lisp program.

Before we do that, let's look at how Lisp handles maths functions, as it is rather unusual. Forget the normal +, −, * and / functions that you may be familiar with.

Lisp not only uses PLUS, DIFFERENCE, TIMES and QUOTIENT, but they also operate in a completely different way.

As you might expect with a powerful list processing language, PLUS will add not just two numbers but complete lists of numbers. For instance, enter:

```
(PLUS 57 328)
(PLUS 5 8 2 6 3)
```

TIMES works in a similar fashion, multiplying all its arguments together. Enter:

```
(TIMES 20 30)
(TIMES 1 2 3 4 5)
```

DIFFERENCE however, can have just two arguments and subtracts the second number from the first:

```
(DIFFERENCE 100 45)
```

QUOTIENT is like BBC Basic's DIV and there is also a corresponding MOD function in Lisp – REMAINDER:

```
(QUOTIENT 26 8)
(REMAINDER 26 8)
```

See if you can work out the result of the following Lisp S-expressions, then enter them at the keyboard and confirm your results:

```
(TIMES (PLUS 3 4) 5)
(SETQ x 48)
(SETQ y 6)
(QUOTIENT x y)
(SETQ n '(56 45))
(PLUS (CAR n) 10)
(PLUS (CAR n) (CAR (CDR n)))
```

That just about sums up

DEFUN of Lisp will now begin...

In Part III of his Lisp series Roland Waddilove shows how to write your own functions

Lisp's maths functions. They are quite primitive and use integers throughout.

Lisp implementations on larger computers don't have this restriction and there are normally many more maths functions.

Floating point arithmetic, SIN, COS, TAN, LOG and so on are commonly found. However, they do require a lot of memory and unfortunately it wasn't possible to implement them on the Electron.

It's now time to write a program and start defining our own functions. For this we'll need to use a built-in Lisp function called DEFUN.

As you know, CAR tells us the first item in a list. Unfortunately, CAR isn't a very good mnemonic so we'll define our own function *first* that does the same job:

```
(DEFUN first (x)
  (CAR x)
)
```

The first word in the definition is DEFUN, informing Lisp that we are about to define a function. This is followed by its name – *first* – and any parameters it may require in brackets.

We are going to pass one parameter, and we'll call it *x*. It's rather like Basic's:

```
DEF FNfirst(x)
```

Lisp doesn't mind whether the parameter is an atom, number or list. In fact, if you wish, it can be different every time you call the

function. In our example we will be passing a list.

The body of the function definition is (CAR x) which simply tells us the first item in the list *x*. Test the function with:

```
(first '(beans on toast))
```

The result is the atom *beans* - the first member of the list (*beans on toast*).

Define the following functions:

```
(DEFUN add1 (n)
  (PLUS n 1)
)

(DEFUN sub1 (n)
  (DIFFERENCE n 1)
)
```

and test them with:

```
(add1 5)
(sub1 5)
(add1 (sub1 5))
```

Last month we looked at a group of commands called predicates. These determine whether something is true or false – T or NIL in Lisp. We'll use these in our next function definitions.

Normally, when we determine whether something is true or false in a program we wish to do something as a consequence – IF something is true THEN do this ELSE do that.

In Lisp we use the built-in function COND. It works in a similar manner to Basic's IF/ THEN/ELSE but is much

more powerful. In general terms the format is:

```
(COND
  (test1 do this)
  (test2 do_that)
  (test3 or_this)
  (test4 or that)
  (...
)
```

Lisp will start with *test1* and evaluate each test in turn. If it finds one that is true it will evaluate the remainder of the line and then skip the rest of the tests and continue after the COND.

For instance, IF *test2* is true THEN it will evaluate *do–that*, whatever it may be.

There can be as many tests as required and the first to evaluate to true ends the COND. It is often useful to offer a default in case none of the tests are true.

For this we use T. As T is always true the test is true and the rest of the line is evaluated. For instance:

```
(COND
  ((ZEROP x) 'zero)
  (T 'non_zero)
)
```

will return *zero* if *x* is zero otherwise *non–zero* will be returned.

We can write a similar function to test for negative numbers:

```
(DEFUN test (x)
  (COND
    ((MINUSP x) 'Negative)
    (T 'Positive)
  )
)
```

Programming

```
FJ ........*........*........*........*........*........*........*........*........(

         Enter and edit Lisp programs in View and save them as normal.
         EXEC them in into Lisp using: (* 'EXEC! name)
         In fact you can set up two function keys to switch between Lisp and View.
         Note that comments like this are ignored.


         *** Recursive program to find the biggest number ***

         (DEFUN input ()
           (PRIN 'Enter two numbers:)
           (biggest (READ) (READ))
         )

         (DEFUN biggest (num1 num2)
           (COND
             ((ZEROP num1) 'second)
             ((ZEROP num2) 'first)
             (T (biggest (sub1 num1) (sub1 num2)))
           )
         )

         (DEFUN sub1 (num)
           (DIFFERENCE num 1)
         )


    ************************************************************************
```

Figure I: A simple recursive program

To see if this works enter:

```
(test 5)
(test -5)
```

Finally, we'll take a brief look at recursion, probably the most important (and difficult to grasp) concept in Lisp programming.

Figure I shows a short Lisp program which will read two numbers from the keyboard and tell us whether the first or the second is biggest.

I'm not holding this up as an example of good Lisp programming, there are much better ways of doing this and it is simply to illustrate recursion. Enter it and type:

```
(input)
```

to run it. Try entering small numbers like 5 and 6 at the prompt and check that the correct answer is printed. You can enter the numbers on separate lines or put them on the same line separated by a space.

What happens with very large numbers? Try entering 1234 and 5678 and see what

happens. You will get a No room error message. All that recursion uses up a lot of memory, so there is a limit to how far you can recurse.

This program is easiest to understand if you start with the last function and read through till you get to the first. In fact you could enter it this way if you wish.

The last function *sub1* simply subtracts one from its argument, *num*. The second function *biggest* expects two numbers *num1* and *num2* and works out which is biggest.

The main body of the function is taken up by COND which has three tests. The first asks if *num1* is zero.

If it is, the second number must be bigger (I'm assuming here that both numbers are positive).

The second test asks if *num2* is zero and if it is, the first number must be biggest.

If neither are zero we come to the default test T. T is always true so Lisp evaluates the rest of the line. This calls *biggest*, but this time

subtracts one from each number first.

So *biggest* keeps on calling itself and each time subtracts one from each number. Eventually one of them will become zero and a result will be found from either the first or second test.

The first function *input*, prints a message on the screen using PRIN and then calls *biggest* passing it two numbers READ from the keyboard.

This short piece of code has all the hallmarks of a Lisp program and is worth studying.

It is far from finished, and you might like to add extra features.

For instance there is nothing to stop you entering two words instead of two numbers (NUMBERP tests for numbers). Also, negative numbers may cause problems (MINUSP tests for negative numbers).

What will happen if both numbers are zero? I'll leave it to you to modify this short program.

Finally, I'll leave you with a more useful Lisp program:

```
(DEFUN DUMP ()
  (PRINC 'Input_filename:)
  (SETQ name (READ))
  (SETQ file (OPEN name 7))
  (LOOP
    (UNTIL (EOF file))
    (PRINTC (READLINE file))
  )
  (CLOSE file)
)
```

This function can be used to display a text file stored on disc. You can use it to look at View files, spooled programs, boot files and so on. Type it in then enter:

```
(DUMP)
```

to run it.

You'll be prompted for a filename, then the text will be read from disc and printed on the screen.

I think you'll agree that it is short, neat and quite easy to understand, even though I have not covered all of the commands used.

● *I hope I've whetted your appetite for Lisp. It's a fascinating language and well worth exploring. Next month we'll be looking at the language Logo, using Acornsoft's Logo cartridge and Turtle Graphics cassette.*

Going up the wall?

I WONDER how many Electron owners when buying their micro, said to their wives, "It'll even help with the housework dear" and have since spent many hours trying to justify that statement.

Well, now you can, with this extremely useful utility to help you calculate how many rolls of wallpaper you will need when decorating a room.

As a bonus, the program also contains some useful routines to use in your own programs, like a box highlighted input function and a rounding up function.

When the program is run an open roll of wallpaper is drawn and you are asked to say if you would like to use decimal or imperial measurements.

Answer with D or I and the program will confirm your choice by printing the word.

Next you will be asked the size of the room that you wish to decorate, both in length and height.

The height is between the skirting board and the ceiling, the length of the walls is the total distance around the room, including doors and windows to allow for any wastage.

When you are working in a room with different ceiling heights measure each height separately and add the totals together.

You will now be asked for the dimensions of the roll of wallpaper.
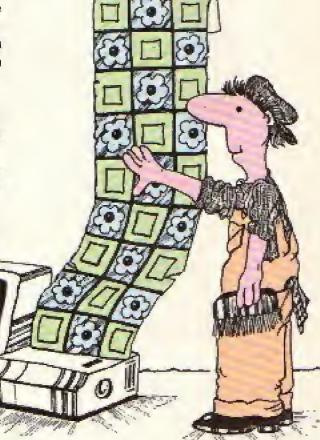
Most rolls sold these days are precut to 10m + 520mm (33ft + 20.5in), so the program will cope with these automaticaly. Just press Return or enter a different value, when prompted.

The total number of rolls required and how many you will have to buy will now be printed.

The total to buy is derived from rounding up the total number of rolls required.
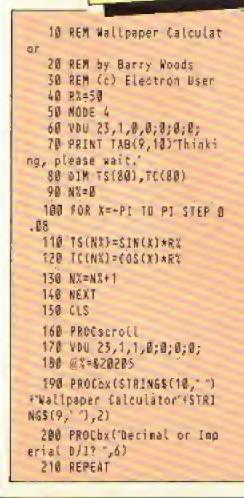
However, please note that this program is intended as a guide, and only provides a rough estimate.

BARRY WOODS presents a utility to help with the decorating

VARIABLES

D	Distance around room
H	Height of walls.
L	Length of roll.
R	How many rolls are needed.
W	Width of roll.
@%	Format numbers to two decimal places.
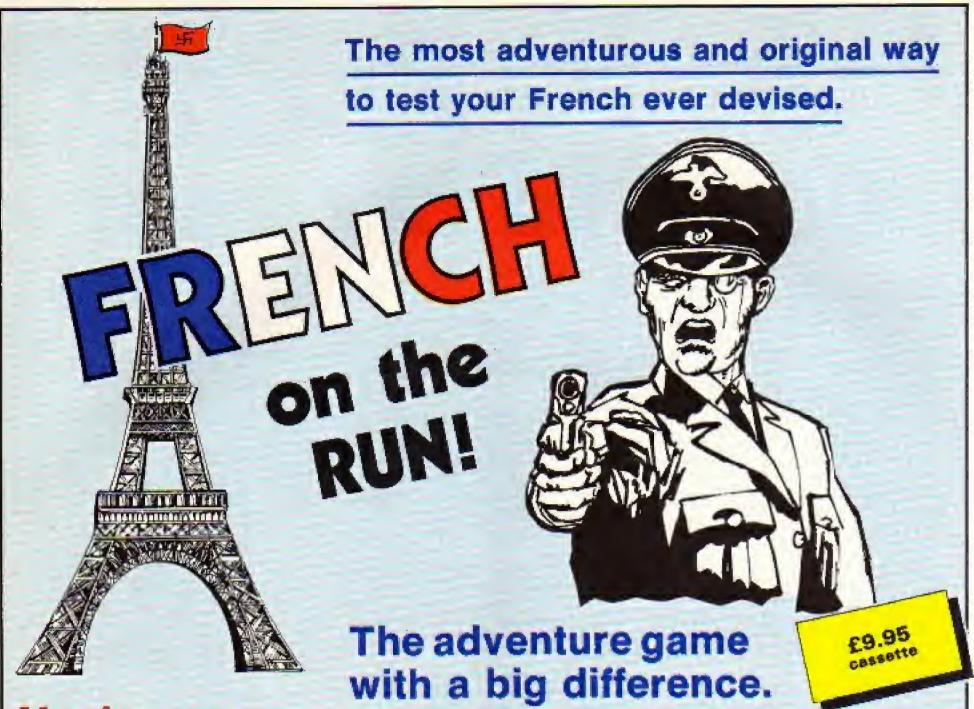
```
   10 REM Wallpaper Calculat
or
   20 REM by Barry Woods
   30 REM (c) Electron User
   40 R%=50
   50 MODE 4
   60 VDU 23,1,0,0;0;0;0;
   70 PRINT TAB(9,10)"Thinki
ng, please wait."
   80 DIM TS(80),TC(80)
   90 N%=0
  100 FOR X=-PI TO PI STEP 0
.08
  110 TS(N%)=SIN(X)*R%
  120 TC(N%)=COS(X)*R%
  130 N%=N%+1
  140 NEXT
  150 CLS
  160 PROCscroll
  170 VDU 23,1,1,0;0;0;0;
  180 @%=&20205
  190 PROCbx(STRING$(10," ")
+"Wallpaper calculator"+STRI
NG$(9," "),2)
  200 PROCbx("Decimal or Imp
erial D/I? ",6)
  210 REPEAT

  220 G=GET AND &DF
  230 IF G=68 MS="metres":n%
=1:PRINT"Decimal" ELSE IF G=
73 MS="feet":PRINT"Imperial"
:m%=0
  240 UNTIL G=73 OR G=68
  250 H=FNbox("Input height
of room in "+MS+" ",8)
  270 D=FNbox("Total length
of walls in "+MS+":",10)
  280 IF D=0 OR H=0 PRINT"
Silly room dimensions":VDU 7
:END
  290 AS="Roll length in "+M
S+" or Return:"
  300 L=FNbox(AS,12)
  310 IF L=0 PROCbx(AS,12):I
F n%=1 PRINT"10m":L=10 ELSE
IF L=0 AND m%=0 PRINT"33ft":
L=33
  320 AS="Roll width in "+MS
+" or Return:"
  330 W=FNbox(AS,14)
  340 IF W=0 PROCbx(AS,14):I
F n%=1 PRINT"520mm":W=0.52 E
LSE IF W=0 AND m%=0 PRINT"20
.5in":W=20.5/12

  350 R=(D/H)/(L/H)
  360 IF R>INT(R) R%=R+1 ELS
E R%=R
  370 PRINT TAB(5,22)"The ro
om requires ";R;" rolls."
  380 @%=0
  390 PRINT TAB(9,24)"Purcha
se ";R%;" rolls"
  400 END
  410
  420 DEF FNbox(m$,Y%)
  430 PROCbx(m$,Y%)
  440 INPUT""T
  450 =ABS(T)
  460
  470 DEF PROCbx(m$,Y%)
  480 xlength=LEN m$*32
  490 y%=1024-Y%*31
  500 MOVE 0,y%+8
  510 DRAW xlength,y%+8
  520 DRAW xlength,y%-40
  530 DRAW 0,y%-40
  540 DRAW 0,y%+8
  550 PLOT 0,4,-12
  560 VDU 5:PRINT m$:VDU 4
  570 PRINT TAB(LEN m$+1,Y%)
;

  580 ENDPROC
  590
  600 DEFPROCscroll
  610 R%=50
  620 VDU 29,100;480;
  630 FOR N%=0 TO 59
  640 PROCplt
  650 NEXT
  660 GCOL 0,0
  670 PROCplt
  680 GCOL 0,1
  690 MOVE -16,-50
  700 DRAW 100,-250
  710 PLOT 85,1000,-50
  720 PLOT 85,1100,-250
  730 PLOT 85,100,-250
  740 VDU 29,200;100;
  750 FOR N%=41 TO 75
  760 PROCplt
  770 NEXT
  780 VDU 29,0;0;
  790 ENDPROC
  800
  810 DEF PROCplt
  820 MOVE TS(N%),TC(N%)
  830 DRAW TS(N%)+1000,TC(N%
)
  840 ENDPROC
```

Don't get your variables in a twist

FRANK MACBETH'S program lets your Electron fingers do the walking

FOR most of us, Basic programs never run successfully first time. I used to overcome the problem using STOP or Escape to halt the program at various points and printing the values of the variables which I thought were causing the problem.

This was very time consuming and trying to remember all the variable names was nearly impossible. The answer was at my finger tips – let my Electron do the job for me.

The machine code when assembled, is stored at memory location &900 on Disc machines using DFS or ADFS and &E00 on Tape machines.

In the last case PAGE must first be set to &1100, but don't worry about setting all this up as the program will do it for you.

To use the utility from within a Basic program insert the line CALL &900, or CALL &E00 depending on which filing system your machine uses.

When this line is executed the screen will clear and the title LIST VARIABLES will appear followed by the question "Numerical values in Hex or Decimal ?". Press H for output in hex or D for decimal output.

The program will ensure that the Caps Lock light is on before you input your choice. When you have made your choice all the active variables will be displayed in the following format:

```
Variable_Name=12  Decimal
Variable_Name=C   Hex
```

The ampersand '&' will not be printed in front of Hex values. The program will also cope with arrays which have up to four subscripts. The output format in this case is:

```
Variable_Name(0,0,0,1)=12
```

or if the array has too many subscripts

```
Variable_Name(More than
        4 subscripts)
```

The program puts the

Electron into paged mode so remember to press the Shift key to view all your variables. When all the variables have been displayed the message "Do you wish A% to Z% ?" will appear.

These are the resident integer variables and are stored in a special section of the Electron's memory. Pressing Y will display them, followed by the message "Any key returns to BASIC".

This is to allow you to examine the value of your variables before handing control back to your Basic program.

To see how the program works we must first examine how Basic stores its variables. A series of linked pointers are used for each letter of the alphabet. For each upper and lower case letter, two memory locations are set aside in page four.

These contain the address, lo-byte first, of the first variable with that particular initial letter.

These are called the base

pointers and start at location &482 for the letter "A" incrementing in steps of two bytes until &4F4 and &4F5 which equate to "z".

The Basic variable, no matter what type – Integer, Floating Point or String – at this address is stored in a standard format which comprises three fields. The length and contents of the fields will vary depending on the type of variable.

The three fields are: The link field, name field, and the value field. The link field contains the address of the next variable with the same first letter as the variable we are looking at.

As this will be a 16 bit address the field will always be two bytes long.

Basic does not store its variables in the zero page of memory so if the hi-byte of the address is zero there are no more variables with the same first letter.

The name field contains the variable name, minus the first letter and terminated by a zero byte.

The value field contains

the actual value of the variable.

The resident integer variables are stored separately. They are allocated four bytes each starting at &404 which is equal to A%.

To fit into a small section of memory the program must use some of the Basic rom routines. To ensure that the Basic rom is paged in the program checks which rom socket Basic is in, (using the osbyte call &BB) compare this with the active language rom, and change it to Basic if necessary.

Using this method it is possible to run the program straight from Disc using a *FILENAME command.

Normally when this type of command is used the filing system rom is paged in, in my case the ADFS. In fact this is not true of the Electron because Basic always occupies rom 11. This routine is provided for compatibility with the BBC micro.

The Basic rom routines

Name	Entry Point	Function
p_char	&B558	This prints the contents of the accumulator using oswrch.
p_space	&B565	This is identical to p-char except that a space character is printed before calling p-char.
p_line	&BC25	Calls osnewl then zeros COUNT.
rdvar	&B32C	Reads a variable value from memory to its relevant accumulator.
cntos	&9EDF	Converts a numerical value into a string which will be in the string accumulator.
p_m	&BFCF	Prints the Ascii string which follows it on to the screen.

Table I: Basic's variable handling routines

and accumulators are listed in Tables I and II with a brief description of how they work, this will help fellow programmers who wish to use them in their own programs.

For further information I suggest you obtain a copy of The Advanced User Guide by Colin Pharo.

When using these routines it is a good idea to save the 'X' and 'Y' registers before entering them as some use these registers

and may not return their original values.

This could be disastrous if like me you tend to use them as counters throughout the program.

The program will also work within a procedure to list any local variables and parameters that have been passed to it.

To aid debugging I have included a checksum, as it is easy to type in LDA instead of LDY. The program will appear to assemble correctly but then refuse to work.

Integer Accumulator	&2A to &2D
Floating Point Accumulator	&2E to &35
String Accumulator	&600 to &6FF

Table II: Basic's main accumulators

At least if your check sum is wrong you can start to check for errors before perhaps locking yourself out of the keyboard.

Tape users should set PAGE to &1100 before loading the object code at &E00, then load their Basic program. Disc users may load the object code after their

Basic program or call it from the disc using *Vars as and when they need it.

One final point if you are using the disc version running at &900 the program runs on into the soft key buffer at &B00 so the contents of the function keys will be lost.

Variable listing

```
  10 REM Print Variables
  20 REM By Frank C Macbeth
  30 REM (c) Electron User
  40 PROCwhere
  50 PROCsetup
  60 PROCass
  70 PROCcheck
  80 END
  90 DEFPROCsetup
 100 IF PAGE=&1100 THEN csum%
=&4105 ELSE csum%=&3995
 110 iA=&2A:zl=&70:lp=&72:x=&
73:y=&74:os=&75:di=&76:ty=&77:
np=&78:el=&7A:el_n=&79:el_t=&7
D:d2=&7F:d_c=&80:d_a=&88
 120 rdvar=&952C:cntos=&9EDF:
p_c=&B558:p_ln=&BC25:p_spc=&B5
65:p_m=&BFC5
 130 res=&404:lk=&402
 140 ENDPROC
 150 DEFPROCass:CLS:VDU19,0,4
,0,0,0:PRINTTAB(13,8)"Please W
ait"
 160 FOR opt%=0 TO 2 STEP 2
 170 P%=s%
 180 [OPT opt%
 190 jsr ba:jsr hex:lda #14:j
sr &FFEE:ldx #&FE
 200 .check inx:inx:lda lk+1,
X:beq ret:jmp pres
 210 .ret cpx #114:bne check
 220 lda #15:jsr &FFEE:jsr p_
m:EQUW &000D:EQUS"Do you wish
AX to ZX ?":nop:jsr &FFE0:cmp
#&59:bne stop
 230 jsr p_ln:ldx#0:stx x
 240 .n_r ldy #0
 250 .n_b lda res,X
 260 sta iA,Y:inx:iny:cpy#4:b
ne n_b
 270 stx x:txa:lsr A:lsr A:cl
c:adc#64:sta y:and #1:sta n_l
 280 jsr p_ln:jmp res_1
 290 .n_l jsr p_spc:lda &1E:c
mp #19:bne n_l
 300 .res_1 lda y:jsr p_c:jsr
 p_m:EQUS "%=":nop:ldy #&40:jsr
 p_c_1:ldx x:cpx#104:bne n_r
 310 .stop jsr p_m:EQUW &000D
:EQUS "Any key returns to BAS1
C":nop:jsr &FFE0:jsr p_ln:rts
 320 .pres sta zl+1:lda lk,X:
sta zl:stx x
 330 .next lda zl:sta np:lda
```

```
zl+1:sta np+1:jsr nam:iny:jsr
fld:dey:dey:jsr v_t:cmp #&FF:b
ne not_ay
 340 jsr aysub:jmp e_m
 350 .not_ay sta iA+2:lda #&3
0:jsr p_c:jsr p_v
 360 .a_m ldy #1:lda (zl),Y:b
ne more
 370 ldx x:jmp ret
 380 .more sta lp:dey:lda (zl
),Y:sta zl:lda lp:sta zl+1:jmp
 next
 390 .psa lda &36:beq fin
 400 ldy #0
 410 .pstra lda &600,Y:jsr p_
c:iny:cpy &36:bne pstra
 420 .fin rts

 430 .v_t lda (zl),Y
 440 cmp #&24:beq sg
 450 cmp #&25:beq ir
 460 cmp #&28:beq ay
 470 lda#5:rts
 480 .sg lda#&&3:rts
 490 .ir lda#4:rts
 500 .ay lda#&FF:rts
 510 .p_v jsr rdvar:cmp#0:beq
 pt
 520 tay
 530 .p_c_1 jsr cntos
 540 .pt jsr psa:rts
 550 .aysub iny:lda(zl),Y
:sec:sbc #3:lsr A:cmp #5:bcc o
k
 560 jsr p_m:EQUS "More than
4 Subscripts":nop:rts
 570 .ok sta di:asl A:sta d2:
dey:dey:dey:jsr v_t:sta ty:cmp
 #5:beq five
 580 lda#4
 590 .five sta os:sta y:iny:i
ny:iny:iny:ldx#0
 600 .s_1 lda(zl),Y:sta d_a,X
:iny:inx:cpx d2:bne s_1
```

```
 610 ldx #0:lda #0:sta el_n:s
ta el_n+1
 620 .s_z sta d_c,X:inx:cpx #
2:bne s_z
 630 .n_el ldy#0
 640 .p_el lda d_c,Y:sta iA:i
ny:lda d_c,Y:sta iA+1:lda #0:s
ta iA+2:sta iA+3:sty el:ldy#&4
0:jsr cntos:jsr psa:lda el:clc
:adc#3:lsr A:cmp di:beq pb
 650 lda #ASC(","):jsr p_c:ld
y el:iny:jmp p_el
 660 .pb jsr p_m:EQUS ")=":no
p:lda y:clc:adc#3:tay:adc(zl),
Y:tay:lda el_n:bne ba_2
 670 lda el_n+1:beq y_corr
 680 .ba_2 lda#0:sta el_t:sta
 el_t+1
 690 .x_os tya:clc:adc os:tay
:bcc ud2
 700 inc zl+1
 710 .ud2 lda el_t:clc:adc#1:
sta el_t:bcc cpr
 720 inc el_t+1
 730 .cpr cmp el_n:bne x_os
 740 lda el_t+1:cmp el_n+1:bn
e x_os
 750 .y_corr jsr fld:lda ty:s
ta iA+2:jsr p_v:lda di:sec:sbc
 #1:asl A:tay
 760 .ud lda d_c,Y:clc:adc#1:
sta d_c,Y:lda d_c+1,Y:adc#0:st
a d_c+1,Y:lda d_a,Y:cmp d_c,Y:
bne x_el
 770 lda d_a+1,Y:cmp d_c+1,Y:
bne x_el
 780 lda#0:sta d_c,Y:sta d_c+
1,Y:dey:dey:cpy #&FE:bne ud
 790 lda np+1:sta zl+1:rts
 800 .x_el lda el_n:clc:adc#1
:sta el_n:bcc rs
 810 inc el_n+1
 820 .rs lda np+1:sta zl+1:js
r nam:jmp n_el
 830 .fld tya:clc:adc zl:sta
iA:lda zl+1:adc#0:sta iA+1:rts
 840 .nam jsr p_ln:lda x:lsr
A:clc:adc #65:ldy#1
 850 .pn_1 lda p_c:iny:lda (n
p),Y:bne pn_1
 860 rts
 870 .hex jsr p_m:EQUS &0C1F0
616:EQUB 1:EQUS"LIST VARIABLES
":EQUW &001F:EQUB 3:EQUS "Numer
```

```
ical values in Hex or Decimal"
:nop:lda#&CA:ldx#&20:jsr &FFF4
 880 .ans jsr &FFE0:cmp#&44:b
eq dec
 890 cmp#&48:bne ans
 900 lda #15:ora#&80:jmp dec_
1
 910 .dec lda &15:and#&7F
 920 .dec_1 sta &15:lda#&2:js
r &FFEE:rts
 930 .ba lda#&B3:ldy#&FF:ldx#
0:jsr &FFF4:txa:cmp &F4:beq b_
d
 940 sta &F4:sta &FE05
 950 .b_d rts
 960 EQUB 0
 970 ]
 980 NEXT
 990 ENDPROC
1000 DEFPROCcheck:P%=P%-1:FOR
t%=s% TO P%:t&=t&+?t%:NEXT:PR
INT"Check Sum=";t&
1010 IF t%<>csum% THEN PRINT"
The Check Sum is NOT correct"'
"It should be ";csum%"Plea
se check your program":ENDPROC
1020 PRINT"Code stored at &"
;"s%;" to &";"P%
1030 PRINT"CALL&";"s%;" to e
xecute"
1040 PRINT "*SAVE Vars ";"s%;
";"P%;" to save object code"
1050 ENDPROC
1060 DEFPROCwhere
1070 AX=0:XX=&70:YX=0:AX=(USR
&FFDA)AND &FE
1080 IF PAGE=&E00 AND AX=1 TH
EN PROCtape
1090 IF PAGE=&1100 THEN s%=&E
00 ELSE s%=&900
1100 ENDPROC
1110 DEFPROCtape
1120 *KEY0FOR I%=TOP TO PAGE S
TEP -1:?(&1100+I%-PAGE)=?I%:NE
XT:PAGE=&1100|MOLD|MRUN|M
1130 *FX138,0,128
1140 END
```

Learning CAN be fun

● Use your Electron to teach and amuse your children at the same time.

● Three packages crammed full of educational programs – and so easy to use!

● Each program has been educationally approved after extensive testing in the classroom.

ONLY
£4.95 tape
£5.95
3½" disc

FUN SCHOOL!
10 programs for 8-12 year olds
BBC Micro & Electron

FUN SCHOOL!
programs for 5-8 year olds
BBC Micro & Electron

Ages 2-5

Alphabet
Colours
Counting
House
Magic Garden
Matchmaker
Numbers
Pelican
Seaside
Snap

PELICAN
Teach your children to cross the road safely at a Pelican crossing

HOUSE
Select the colours to draw a house – hours of creative entertainment

Ages 5-8

Balance
Castle
Derrick
Fred's Words
Hilo
Maths Test
Mouser
Number Signs
Seawall
Super Spell

NUMBER SIGNS
Provide the correct arithmetic sign and aim to score ten out of ten

BALANCE
Learn maths the fun way. Type in the answer to balance the scales

Ages 8-12

Anagram
Codebreaker
Dog Duck Corn
Guessing
Hangman
Maths Hike
Nim
Odd Man Out
Pelmanism
Towers of Hanoi

HANGMAN
Improve your child's spelling with this fun version of the popular game

ODD MAN OUT
Find the word that does not fit – before your time runs out

TO ORDER TURN TO THE FORM ON PAGE 53

THE BIG SQUEEZE

Bill Trevelyan discusses text compression techniques in Part II of his adventure programming series

IN the last article we discussed ways of storing text (which makes up a large proportion of any adventure program) open to the programmer embarking on the lengthy task of writing a game in Basic, with perhaps a bit of machine code thrown in here and there.

The most promising method is to set HIMEM at some convenient figure (say &2800), while &E00 to HIMEM holds the Basic program. HIMEM to &5FFF (in Mode 6) is reserved for the database.

Text, such as location descriptions, is stored as a series of strings made up of Ascii codes.

Each string is terminated by a carriage return (&0D) and identified by a serial number. This enables the address of the start of the string to be read from a list of two byte addresses also stored in the database.

It is best to store location descriptions and the shorter messages as two separate groups. Each should have its own address list, largely for convenience in writing the program.

The technique can be improved, especially when memory is tight – but only at the cost of more elaborate programs to insert text into the database, and to recover it as and when required.

The first task is to get rid of the extra spaces which were inserted in order to produce an attractive appearance on the screen.

Enter and run Program I. Text is pulled out of memory one character at a time using:

```
150 cS=CHRS?IX
```

In Basic this is a slow business, but it is easily and rapidly executed in machine code.

If c$ is a space and the line extends more than three quarters across the screen a new line is printed. This simple formatting routine works perfectly provided you don't cram your program full of 10 letter words or hyphenated phrases.

Note that if the program to insert text into memory also operates one character at a time, as in Program II, the 235 character (approx-imate) limit on a description or message no longer applies.

The same effect can be achieved in another way. The passages of text in adventure games are made up of 52 letters, mostly lower case, the space character and a few punctuation marks.

There are, however, 96 Ascii codes and we are at liberty to re-define those we don't normally use to stand for groups of letters, words or phrases.

Provided they occur in our text fairly often, significant savings in memory can be made – this is the approach used in Woodbury End.

In Program III text held in the database is examined character by character. If "#" is found the phrase "You are " is printed, and the occurrence of "=" invokes a line feed.

The procedure PROCentertext in Program IV is a definite help in correctly placing the line feed symbol when text is being inserted into the database.

This method of selecting and printing a piece of text from the database is very easy to translate into machine code. The routine

```
 10 REM Program I
 20 locS="in an octagonal
room bathed in a shadow-free
radiance by concealed light
ing. Masterpieces of modern
art line the walls. Archways
lead off to E and W. N is a
close-fitting door. An orna
te chair is nearby."
 30 REM Text placed in mem
ory starting at &4000
 40 $&4000=locS
 50 REM List of addresses
at &3100
 60 !&3100=&40CD4000
 70 PROCL(0)
 80 END
 90
 100 DEFPROCL(locX)
 110 TX=!(&3100+2*locX) AND
&FFFF
 120 EX=!(&3100+2*locX+2) A
ND &FFFF
 130 PRINT"You are ";
 140 FOR IX=TX TO EX-2
 150 cS=CHRS?IX
 160 IF cS=" " AND COUNT>30
PRINT ELSE PRINT cS;
 170 NEXT
 180 PRINT
 190 ENDPROC
 200 REM Time taken was 1.0
s
 210 REM Length of text=204
```

Program I

```
 10 REM Program II
 20 locS="#in an octagonal
room bathed in=a shadow-fre
e radiance by concealed=ligh
ting. Masterpieces of modern
art=line the walls. Archway
s lead off to E=and W. N is
a close-fitting door. An=orn
ate chair is nearby."
 30 REM Text placed in mem
ory starting at &4000
 40 $&4000=locS
 50 REM List of addresses
at &3100
 60 !&3100=&4000
 70 PROCL(0)
 80 END
 90
 100 DEFPROCL(locX)
 110 TX=!(&3100+2*locX) AND
&FFFF
 120 IX=0
 130 REPEAT
 140 cS=CHRS(IX?TX)
 150 ON INSTR("#=",cS)GOSUB
200,210 ELSE PRINT cS;
 160 IX=IX+1
 170 UNTIL ASCcS=&0D
 180 PRINT
 190 ENDPROC
 200 PRINT"You are ";:RETUR
N
 210 PRINT:RETURN
 220 REM Time taken was 1.6
seconds
```

Program II

Program III

```
10 REM Program III
20 loc$="in an octagonal
room bathed in a shadow free
radiance by concealed light
ing. Masterpieces of modern
art line the walls. Archways
lead off to E and W. N is a
close fitting door. An orna
te chair is nearby."
30 REM Text placed in mem
ory starting at &4000
40 $&4000=loc$
50 REM List of addresses
at &3100
60 !&3100=&4000
70 PROCL(0)
80 END
90
100 DEFPROCL(locX)
110 TX=!(&3100+2*locX) AND
&FFFF
120 PRINT"You are ";
130 IX=0
140 REPEAT
150 c$=CHR$(IX?TX)
160 IF c$=" " AND COUNT>30
PRINT ELSE PRINT c$;
170 IX=IX+1
180 UNTIL ASCc$=&0D
190 PRINT
200 ENDPROC
210 REM Time taken=1.5 s
220 REM Length of text=204
```

Program V

```
10 REM Program V
20 HIMEM=&2800
30 PROCinit
40 INPUT"Assemble code
(Y/N)?"a$
50 IF a$="Y" OR a$="y" TH
EN PROCassemble
60 PRINT"Press SPACE to
continue"'
70 REPEAT:G=GET:UNTIL G=3
2
80 PROCL(0)
90 PROCm(0)
100 END
110
120 DEFPROCassemble
130 code=&2800:osasci=&FFE
3
140 list=&70:addr=&72:offs
et=&74
150 FOR pass=0 TO 2 STEP 2
160 PX=code
170 [OPT pass
180 .start
190 STA offset
200 STX list:STY list+1
210 LDA offset
220 ASL A   \Multiply numb
er by 2
230 TAY
240 LDA #0  \Carry set if
number>128
250 ADC list+1
260 STA list+1
270 LDA (list),Y \address
of text,low byte
280 STA addr
290 INY
300 LDA (list),Y \address,
high byte
310 STA addr+1
320 .read
330 LDY #0
340 .step
350 LDA (addr),Y
360 CMP #&00
370 BEQ fin
380 CMP #&3D    \the = sig
n
390 BNE print
400 LDA #&0D
410 .print
420 JSR osasci
430 INY
440 JMP step
450 .fin
460 JMP osasci
470 ]
480 NEXT pass
490 :
500 PRINT"Code ends at &"
;~PX
510 ENDPROC
520 :
530 DEFPROCinit
540 RESTORE 750
550 READ loc$
560 $&4000=loc$
570 !&2900=&4000
580 READ mess$
590 $&3000=mess$
600 !&2A00=&3000
610 ENDPROC
620 :
630 DEFPROCL(locX)
640 A%=locX
650 XX=&00:YX=&29
660 CALL &2800
670 ENDPROC
680 :
690 DEFPROCm(messX)
700 A%=messX
710 XX=&00:YX=&2A
720 CALL &2800
730 ENDPROC
740 :
750 DATA "You are in an oc
tagonal room bathed in=a sha
dow-free radiance by conceal
ed=lighting. Masterpieces of
modern art=line the walls.
Archways lead off to E=and W
. N is a close-fitting door.
An=ornate chair is nearby."
760 DATA "==A strange rust
ling noise can be=heard.=""
```

Program IV

```
10 REM Program IV
20 HIMEM=&2800:SX=HIMEM
30 INPUT"List to start a
t &"list$
40 listX=EVAL("&"+list$)
50 INPUT"Text to start a
t &"text$
60 textX=EVAL("&"+text$)
70 countX=0:JX=0
80 REPEAT
90 FOR IX=0 TO &1FF:IX?SX
=0:NEXT
100 PROCentertext
110 PRINT''"Length of entr
y: ";NX;" chars"
120 PROCenteraddress
130 PROCentercode
140 PRINT'"Next coded te
xt at &";~(textX+JX)
150 countX=countX+1
160 PRINT'"Another entry
(Y/N)?"
170 A$=GET$
180 UNTIL INSTR("Yy",A$)=0
190 VDU7:END
200 :
210 DEFPROCentertext
220 CLS:PRINT"Enter text
:";""
230 NX=0
240 REPEAT
250 GX=GET
260 PRINT CHR$(GX);
270 NX?SX=GX
280 IF GX=&3D OR GX=&0D TH
EN PRINT
290 IF GX=&7F NX=NX- ELSE
NX=NX+1
300 UNTIL GX=&0D
310 ENDPROC
320 :
330 DEFPROCentercode
340 iX=0:jX=0
350 F=TRUE
360 FOR IX=0 TO NX
370 byteX=IX?SX
380 QX=INSTR(" etaonri",CH
R$(byteX))
390 IF QX PROCcode:F=NOT F
:GOTO 410
400 PROCclear
410 NEXT IX
420 ENDPROC
430 :
440 DEFPROCcode
450 IF F THEN iX=QX+7:ENDP
ROC
460 IF NOT F THEN jX=QX+7
470 PROCout
480 ENDPROC
490 :
500 DEFPROCclear
510 IF NOT F THEN 560
520 iX=byteX DIV 16
530 jX=byteX MOD 16
540 PROCout
550 ENDPROC
560 jX=byteX DIV 16
570 PROCout
580 iX=byteX MOD 16
590 ENDPROC
600 :
610 DEFPROCout
620 JX?textX=16*iX+jX
630 JX=JX+1
640 ENDPROC
650 :
660 DEFPROCenteraddress
670 TX=textX+JX
680 ?(listX+2*countX)=TX M
OD 256
690 ?(listX+2*countX+1)=TX
DIV 256
700 ENDPROC
```

in Program V will print out a 200 character location description in 0.2 sec.

The machine code is only 31 bytes long and can be tucked away at the beginning of the database. It is called from one of two simple Basic procedures, one for location descriptions and one for short messages.

When a machine code routine is called the integer A% is transferred to the accumulator, while X% and Y% go into the X and Y registers respectively.

This is the way the procedures pass the number of the desription or message, as well as where the appropriate list of addresses start. This is probably the best method to use as it is simple, fast and economical.

There are endless schemes for cramming a quart of text into a pint pot – text compression, to use the jargon. These are based on the realisation that a byte can assume 256 different configurations, of which only 96 are used for Ascii codes.

As we have seen, the adventure writer needs about 64 of these. In par-

ticular, all codes above 128 (&80 in hex) are unused.

In adventure programs these codes may be appropriated to stand for words, groups of two or three letters, and so on.

Unfortunately, even if we confine ourselves to lower case letters there are 576 possible two-letter combin-
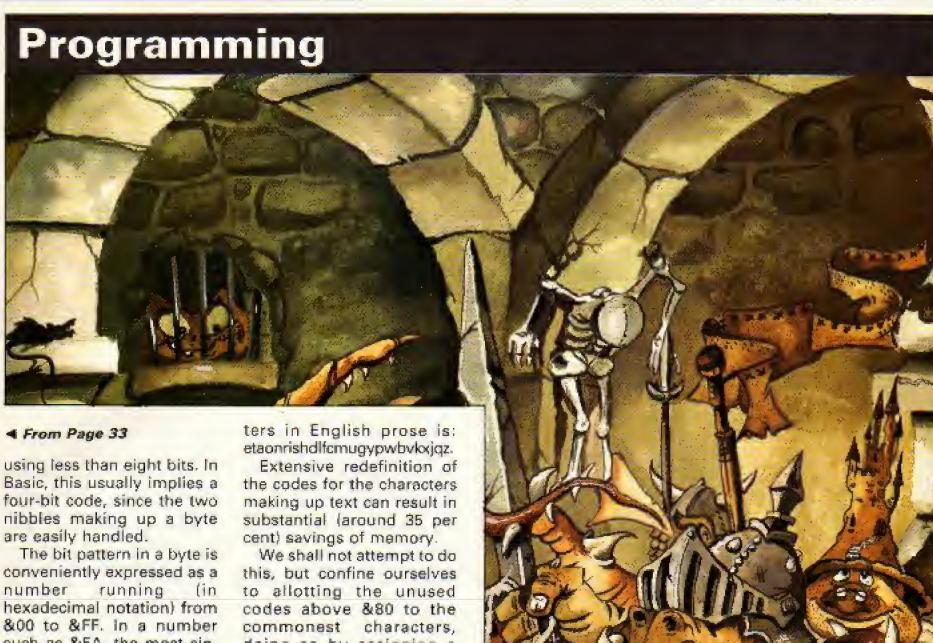
ations and many more of three letters. So a selection of those most frequently used must be made. All this leads to complex programming and routines which are very slow in Basic.

The fundamental unit of memory as far as the programmer is concerned is the byte, though occasionally

we concern ourselves with the state of individual bits, more so in machine code than in Basic.

Instead of one byte representing more than one letter, we can restrict ourselves to handling single characters which are coded

Programming

using less than eight bits. In Basic, this usually implies a four-bit code, since the two nibbles making up a byte are easily handled.

The bit pattern in a byte is conveniently expressed as a number running (in hexadecimal notation) from &00 to &FF. In a number such as &5A, the most significant nibble is &5 and the least significant &A.

Of course, we can't manipulate four-bit nibbles as such in a program but we can store the value &5 in $i\%$ and &A in $j\%$. Then the value of the original byte can be calculated and placed in a third variable with:

    byte%=16*i%+j%

Conversely, the nibble values are obtained with:

    i%=byte% DIV 16
    j%=byte% MOD 16

Four bits enable 2, 4 or 16 items to be coded. In the case of our adventure game text, we need some 64 codes.

It follows that a four-bit code can be only partial, with most characters still requiring a full byte.

The four-bit patterns will be reserved for the characters which occur most frequently. Two of these will certainly be the space character and the letter e.

If you are so minded, you could write a program to establish a pecking order for your particular text.

In general though, the order of frequency for letters in English prose is: etaonrishdlfcmugypwbvkxjqz.

Extensive redefinition of the codes for the characters making up text can result in substantial (around 35 per cent) savings of memory.

We shall not attempt to do this, but confine ourselves to allotting the unused codes above &80 to the commonest characters, doing so by assigning a nibble value in the range &8 to &F (8 - 15) to space, e, t, a, o, n, r and i.

Take the word Electron. In Ascii this requires eight bytes holding the following hex values:

    45 6C 65 63 74 72 6F 6E

but when our four-bit code is substituted for the common letters it becomes:

    45 6C 09 63 8A 0E 0C 8D

and when these are collected in pairs to form bytes which can be stored in memory we arrive at:

    45 6C 96 3A EC 08

which corresponds to a 25 per cent saving in memory usage. Note that a zero had to be stuck on the end to make up an even number of nibbles.

All very well you may say, but when this is decoded how do we know that the C in byte two is the least significant nibble of an Ascii code, while that in byte five stands for the letter n?

Any nibble less than eight must be part of an Ascii code, since we are using only values of &8 to &F to code the special (most common) characters. So, if we have an unpaired nibble less than eight, the next nibble must be its partner in an Ascii code.

Program VI decodes the data. A byte is taken from the coded string and the values of the two incoming nibbles are stored in the variables $nib\%(1)$ and $nib\%(2)$. Outgoing nibbles are put in two other variables, $i\%$ and $j\%$.

When a pair of nibbles has been collected, the byte $C\%$ is calculated as $16*i\%+j\%$ and sent to be printed.

A flag $F\%$ holds information on the buffer variables $i\%$ and $j\%$, being TRUE

```
   10 REM Program VI
   20 CLS:INPUT"List to sta
rt at: &"list$:list%=EVAL("&
"+list$)
   30 INPUT''"Enter serial
number of text: "L%:PRINT'"
   40 DIM nib%(2)
   50 PROCdecode(L%)
   60 END
   70 :
   80 DEFPROCdecode(loc%)
   90 text%=!(list%+2*loc%)
AND &FFFF
  100 J%=0:F%=TRUE
  110 REPEAT
  120 byte%=J%?text%
  130 nib%(1)=byte% DIV 16
  140 nib%(2)=byte% MOD 16
  150 FOR n%=1 TO 2
  160 IF F% AND nib%(n%)>=8
PROCp
  170 IF F% AND nib%(n%)<8 i
%=nib%(n%):F%=FALSE:GOTO 190
  180 IF NOT F% j%=nib%(n%):
PROCpp:F%=TRUE
  190 NEXT
  200 J%=J%+1
  210 UNTIL C%=&0D
  220 ENDPROC
  230 :
  240 DEFPROCp
  250 C$=MID$("1234567 etaon
ri",nib%(n%),1)
  260 PRINT C$;
  270 ENDPROC
  280 :
  290 DEFPROCpp
  300 C%=16*i%+j%
  310 IF C%=&0D OR C%=&30 TH
EN PRINT ELSE PRINT CHR$C%;
  320 ENDPROC
```

Program VI

Programming

if both have been filled and FALSE if i% only has been filled.

For our six byte line of code (the string "Electron") let us take the first byte, &45. The &4 goes into nib%(1) and &5 into nib%(2).

Since nib%(1) is less than eight, the value &4 is assigned to i% and F% set to FALSE.

Now nib%(2) holds &5 and since F% is FALSE this goes to j% as the second nibble of an Ascii code, F% reverting to TRUE as a sign to send the byte &45 to be printed as the letter E.

With the next byte, again the value in nib%(1) goes into i% and nib%(2) into j%. As F% is FALSE, the &6C is printed as the letter l.

With byte three we have nib%(1) containing &9 and nib%(2) with &6. As F% is

TRUE nib%(1) holds the code for a special character, in this case e, which is printed.

Looking at the value in nib%(2) we find a value less than eight and the flag at TRUE, so it is held in i% until a partner is found (the first nibble of the next byte). And so on for the rest of the string.

Note that if you start decoding at byte three you would get "ectron" but if you start at the next byte you would get a colon printed out (Ascii code &3A).

The coding program works on a similar principle. A piece of text is entered (use extra spaces for formatting or "=" as a line feed), and held in a 256 byte buffer at &2800 to &28FF.

The string when complete is examined byte by byte and the two nibbles of Ascii

codes for special characters replaced by one.

The outgoing nibbles are collected in pairs before being sent as completed bytes to be stored in the database, together with a list of addresses for the decoder to start at the correct place.

About 30 per cent saving of memory is possible, depending on the make-up of the text.

The decoding program isn't too long, and in practice would be shortened by use of multiple statement lines, elimination of surplus spaces and so on.

The snag? As ever with Basic, it is speed – Program IV takes five seconds to print out a 200 word text. However, it translates quite easily into machine code.

If four-bit codes sound a mite forbidding, there is an alternative solution. It is

based on the principle that Ascii codes don't use bit 7, that is, don't exceed &7F.

We can therefore use this bit as a flag – a signal that something is or is not to be done.

Here, when a space character occurs in a piece of text, it is erased and the flag bit set in the Ascii code for the succeeding character.

The coded string is shorter by about 15 per cent (average word-length six).

The decoding program is simple. Each byte is examined in turn to see if it exceeds &80; if it does, it is ANDed with &7F to clear the flag bit and attached to a space character before being printed.

● *There's plenty here to get your teeth into and should keep you occupied till next month when we'll see how to write a full adventure.*
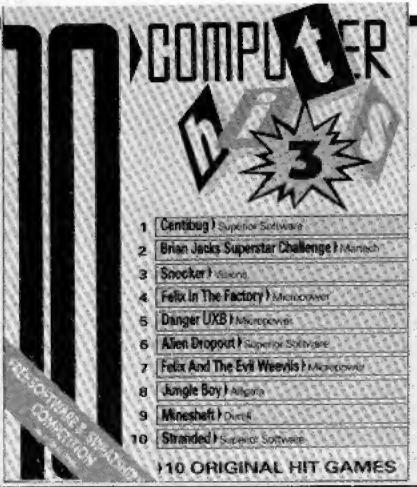
Kaleidoscope

JOANNE STEVENS revives a popular childhood toy

ONE of the most absorbing toys I played with as a child was a kaleidoscope.

The shapes, revolving round in a drum and mirrored manyfold, made myriads of patterns which, no matter how many times you used the instrument, never exactly duplicated themselves.

The outstanding graphics of the Electron seemed to be a perfect base from which to replicate my old childhood toy.

The theory behind the program is exactly the same as in the original object: A pattern is drawn within a confined space on the screen, then mirrored around the rest of the area. The result is a graphic display of constantly changing shape and colour.

There are two versions of the program, but both work in exactly the same way. Program I is a Basic listing for Kaleidoscope and Program II is the same program written in assembly language.

In this particular case there isn't a great difference in performance between Basic and machine code. That's because most of the

time the machine code is simply hanging around waiting for the operating system to finish drawing a line.

It is interesting, however, to see how a simple Basic listing may be converted to assembly language.

When either version is run, you'll first be asked to input the mode. This can be any graphics mode. The origin is then moved to the centre of the screen and a REPEAT/UNTIL loop is entered.

The kaleidoscope works by picking two points in just one sector and drawing a line between them.

The coordinates are incremented, a new line is drawn and the process is repeated a random number of times.

The line is reflected about the x and y axes in the same way that the mirrors reflect the shapes in the kaleidoscope tube.

Perhaps the most interesting aspect of the program is how such intricate patterns can be produced by such a short listing. Type it in then sit back and watch the multitude of patterns swirling round the screen.

```
 10 REM Kaleidoscope
 20 REM By Joanne Stevens
 30 REM (c) Electron User
 40 MODE 6
 50 PRINT"KALEIDOSCOPE"
 60 PRINT "============"
 70 INPUT"Which MODE ";mo
de
 80 MODE mode:*FX16
 90 VDU23,1;0;0;0;0;
100 VDU 29,640;512;
110 REPEAT
120 GCOL 0,RND(8)-1
130 R=RND(400)
140 a=RAD(RND(45))
150 A=R*SIN(a)
160 B=R*COS(a)
170 R=RND(400)
180 a=RAD(RND(45))
190 C=R*SIN(a)
200 D=R*COS(a)
210 X=8*(RND(2)-1)
220 Y=(X EOR 8)/2
230 FOR J=1 TO RND(25)
240 MOVE A,B:DRAW C,D
250 MOVE B,A:DRAW D,C
260 MOVE B,-A:DRAW D,-C
270 MOVE A,-B:DRAW C,-D
280 MOVE -A,-B:DRAW -C,-D
290 MOVE -B,-A:DRAW -D,-C
300 MOVE -B,A:DRAW -D,C
310 MOVE -A,B:DRAW -C,D
320 A=A+X:B=B+Y
330 C=C+X:D=D+Y
340 NEXT
350 UNTIL FALSE
```

Program I

```
 10 REM Kaleidoscope
 20 REM By Joanne Stevens
 30 REM (c) Electron User
 40 MODE 6
 50 PRINT"KALEIDOSCOPE"
 60 PRINT "============"
 70 INPUT"Which MODE ";no
de
 80 PROCassemble:*FX16
 90 MODE mode:*FX16
100 VDU23,1;0;0;0;0;
110 VDU 29,640;512;
120 REPEAT
130 GCOL 0,RND(8)-1
140 RX=RND(400):a=RAD(RND(
45)):!AX=RX*SIN(a):!BX=RX*CO
S(a)
150 RX=RND(400):a=RAD(RND(
45)):!CX=RX*SIN(a):!DX=RX*CO
S(a)
160 ?count=RND(25)
170 ?xstep=8*(RND(2)-1):?y
step=(?xstep EOR 8)DIV 2
180 CALL start
190 UNTIL FALSE
200
210 DEF PROCassemble
220 os=!&20E AND &FFFF
230 AX=&68:BX=&6A:CX=&6C:D
```

```
X=&6E
240 aX=&60:bX=&62:cX=&64:d
X=&66
250 xstep=&90:ystep=&91:co
unt=&92
260 FOR pass=0 TO 2 STEP 2
270 PX=&900
280 [OPT pass
290 .start
300 SEI
310 LDA AX:EOR #&FF:STA aX
:LDA AX+1:EOR #&FF:STA aX+1
\aX=-AX
320 INC aX:BNE ok1:INC aX+
1
330 .ok1
340 LDA BX:EOR #&FF:STA bX
:LDA BX+1:EOR #&FF:STA bX+1
\bX=-BX
350 INC bX:BNE ok2:INC bX+
1
360 .ok2
370 LDA CX:EOR #&FF:STA cX
:LDA CX+1:EOR #&FF:STA cX+1
\cX=-CX
380 INC cX:BNE ok3:INC cX+
1
```

Program II

Graphics

```
 390 .ok3
 400 LDA D%:EOR #&FF:STA d%
:LDA D%+1:EOR #&FF:STA d%+1
\d%=-D%
 410 INC d%:BNE ok4:INC d%+
1
 420 .ok4
 430 \
 440 \ x,y
 450 LDA #25:JSR os:LDA #4:
JSR os
 460 LDA A%:JSR os:LDA A%+1
:JSR os
 470 LDA B%:JSR os:LDA B%+1
:JSR os
 480 LDA #25:JSR os:LDA #5:
JSR os
 490 LDA C%:JSR os:LDA C%+1
:JSR os
 500 LDA D%:JSR os:LDA D%+1
:JSR os
 510
 520 \ y,x
 530 LDA #25:JSR os:LDA #4:
JSR os
 540 LDA B%:JSR os:LDA B%+1
:JSR os
 550 LDA A%:JSR os:LDA A%+1
:JSR os
 560 LDA #25:JSR os:LDA #5:
JSR os
 570 LDA D%:JSR os:LDA D%+1
```

```
:JSR os
 580 LDA C%:JSR os:LDA C%+1
:JSR os
 590
 600 \ y,-x
 610 LDA #25:JSR os:LDA #4:
JSR os
 620 LDA B%:JSR os:LDA B%+1
:JSR os
 630 LDA a%:JSR os:LDA a%+1
:JSR os
 640 LDA #25:JSR os:LDA #5:
JSR os
 650 LDA D%:JSR os:LDA D%+1
:JSR os
 660 LDA c%:JSR os:LDA c%+1
:JSR os
 670
 680 \ x,-y
 690 LDA #25:JSR os:LDA #4:
JSR os
 700 LDA A%:JSR os:LDA A%+1
:JSR os
 710 LDA b%:JSR os:LDA b%+1
:JSR os
 720 LDA #25:JSR os:LDA #5:
JSR os
 730 LDA C%:JSR os:LDA C%+1
:JSR os
 740 LDA d%:JSR os:LDA d%+1
:JSR os
 750
 760 \ -x,-y
 770 LDA #25:JSR os:LDA #4:
JSR os
```

```
 780 LDA a%:JSR os:LDA a%+1
:JSR os
 790 LDA b%:JSR os:LDA b%+1
:JSR os
 800 LDA #25:JSR os:LDA #5:
JSR os
 810 LDA c%:JSR os:LDA c%+1
:JSR os
 820 LDA d%:JSR os:LDA d%+1
:JSR os
 830
 840 \ -y,-x
 850 LDA #25:JSR os:LDA #4:
JSR os
 860 LDA b%:JSR os:LDA b%+1
:JSR os
 870 LDA a%:JSR os:LDA a%+1
:JSR os
 880 LDA #25:JSR os:LDA #5:
JSR os
 890 LDA d%:JSR os:LDA d%+1
:JSR os
 900 LDA c%:JSR os:LDA c%+1
:JSR os
 910
 920 \ -y,x
 930 LDA #25:JSR os:LDA #4:
JSR os
 940 LDA b%:JSR os:LDA b%+1
:JSR os
 950 LDA A%:JSR os:LDA A%+1
:JSR os
 960 LDA #25:JSR os:LDA #5:
JSR os
 970 LDA d%:JSR os:LDA d%+1
```

```
:JSR os
 980 LDA C%:JSR os:LDA C%+1
:JSR os
 990
1000 \ -x,y
1010 LDA #25:JSR os:LDA #4:
JSR os
1020 LDA a%:JSR os:LDA a%+1
:JSR os
1030 LDA B%:JSR os:LDA B%+1
:JSR os
1040 LDA #25:JSR os:LDA #5:
JSR os
1050 LDA c%:JSR os:LDA c%+1
:JSR os
1060 LDA D%:JSR os:LDA D%+1
:JSR os
1070
1080 .CLC
1090 LDA A%:ADC xstep:STA A
%:LDA A%+1:ADC #0:STA A%+1
1100 LDA B%:ADC ystep:STA B
%:LDA B%+1:ADC #0:STA B%+1
1110 LDA C%:ADC xstep:STA C
%:LDA C%+1:ADC #0:STA C%+1
1120 LDA D%:ADC ystep:STA D
%:LDA D%+1:ADC #0:STA D%+1
1130
1140 DEC count:BNE exit
1150 JMP start
1160 .exit
1170 RTS
1180 ]
1190 NEXT
1200 ENDPROC
```

Disabler

QUITE a lot of Electron software insists that you disable the Plus 1 and 3 before you load it. This can be a nuisance so Jason Fisher has written a superb utility to switch off both these units.

When the program is run, a machine code file called DISABLE is saved to disc. When you want to disable the Plus 1 and Plus 3, for instance just before you play a game, all you have to do is insert the disc and load and run the machine code program with *DISABLE.

```
   1REM Disabler Plus 1,3
   2REM by Jason Fisher
   3MODE6
   4FORD=0TO2STEP2:P%=&5E0
0:[OPT D:LDX#0:.loop:STA&70
:STY&71:STX&72:LDA#138:LDY
data,X:LDX#0:JSR&FFF4:LDA&7
0:LDY&71:LDX&72:INX:CPX=20:
BNE loop:RTS:.data:]:NEXT
   5DATA42,66,65,46,13,80,
65,71,69,61,38,55,70,48,48,
13,82,85,78,13
   6FORD=0TO19:READX:?(D+P

%)=X:NEXT:XX=PAGE:REPEAT:XX
=XX+1:UNTIL?XX=224:XX=XX+1:
FORE%=0TO TOP-PAGE STEP4:I%
!&5E00=I%!XX:NEXT:OSCLI FAD
ES":OSCLI'DIR
   7*SAVE DISABLE 5E00 5F5
0 5E00 5E00
   8END
   9*KEY10*FX163,128,1:M?&
212=&06:M?&213=&71:M?&2AC=0
:M
  10?&DF4=&9C:CALL!-4
```

Compressor

AS an aid to writing more compact programs we present a superb 10-line program compactor from G.Shaw.

To use it, first enter the listing then save it. Load in the program that you wish to compress, set PAGE to &5A00 and reload the

compact utility.

When this is done run the program and enter the old value of PAGE (normally &E00 or &1D00). Do not precede the hex number with an & symbol when entering it.

The routine is quite slow, taking approximately 15

seconds per 1000 bytes, but you only have to use it once so this should not be a problem.

The program removes all single line rems and excess spaces, but must not be used for assembly language listings as it will not work correctly.

```
   1 INPUT"Old value of PA
GE="A$:Z%=EVAL("&"+A$):s=0
:p=Z%:A=Z%:L=0:sw=0:t=0:sw1
=0:n=0:*K.0:NL.:M
   2 REPEAT:?A?3:s=0:FORB
%=A TO A+3:?p=?B%:p=p+1:NEX
TB%:n=p-1:B%=A+3:REPEAT:B%=
B%+1:UNTIL?B%<>32:IF?B%=220
:sw1=1
   3 FORB%=A+4TO(A+L)-1:IF
?B%=34ANDsw=1:sw=0:GOTO6
   4 IF?B%=34:sw=1
   5 IFsw=1ORsw1=1 GOTO7
   6 IF?B%=32:s=s+1:GOTO8
   7 ?p=?B%:p=p+1
   8 NEXTB%:?n=(L-s):(?n?)
=244THENp=n-3:s=[
   9 A=A+L:t=t+s:IFsw=1TH
ENsw1=0
  10 UNTIL?A=13ANDA?1=255:
?p=13:p2?=255:PRINT"Space s
aved:";t:PAGE=Z%:*FX135,0,3
28
```

Magic Forest
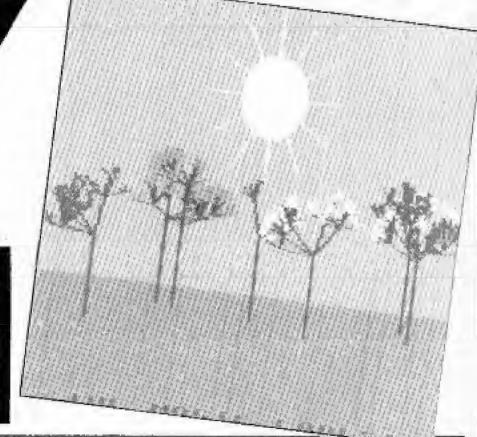
IN the June 1987 issue of *Electron User* Mike Cook showed how recursion could be employed to produce a Dragon curve. Bernard Fox has used a similar recursive technique to produce a superb graphic picture of a Magic Forest.

The continuously growing trees are drawn using recursion and slowly fill the screen. The program never ends, so press Escape when the forest becomes too densely populated and re-run it to produce a different one.

```
   1 REM Magic Forest
   2 MODE2:VDU23,1,0;0;0;0;
:COLOUR132:CLS:VDU28,0,31,19
,22:COLOUR130:CLS:PROCsun:VD
U26:COLOUR1:PRINTTAB(2,31)"T
HE MAGIC FOREST":VDU5
   3 REPEAT:XX=RND(1000)+14
0:YX=RND(80)+200:GCOL0,0:CX=

RND(7):MOVEXX,YX:PLOT1,0,200
:PROCbranch(100,XX,YX+200):U
NTIL0
   4 DEFPROCsun:VDU29,640:8
00;:GCOL0,3:FORR%=-100 TO 10
0 STEP 4:MOVER%,SQR(100^2-R%
^2):DRAWR%,-SQR(100^2-R%^2):
NEXT:FORs=0 TO 2*PI STEP PI/

6:MOVE0,0:DRAW200*COSs,200*S
INs:NEXT:ENDPROC
   5 DEFPROCbranch(J%,XX%,Y
Y%):LOCALB%
   6 IF J%<13 ENDPROC
   7 FOR B%=0 TO 4
   8 IF J%<40 GCOL0,C% ELSE
GCOL0,0

   9 A%=RND(180):a=RAD(A%):
MOVEXX%,YY%:DRAWXX%+J%*COSa,
YY%+J%*SINa:PROCbranch(J%/2,
XX%+J%*COSa,YY%+J%*SINa):NEX
T
  10 GCOL0,RND(7):PLOT69,RN
D(1279),RND(200)+60:ENDPROC
```
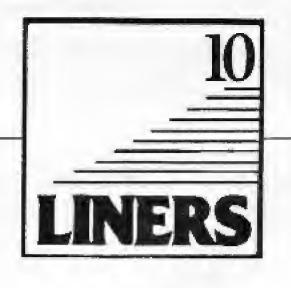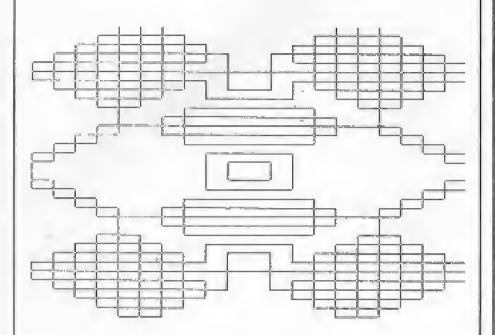
10 LINERS

Racing rectangles

THIS short graphics program by Stephen Smith produces a never ending series of patterned lines on the screen.

A rectangle is drawn in the centre and the four sides expand rapidly outwards toward the edge of the screen. As they hit the sides they bounce inwards, contracting until they reach the centre, starting the whole process off again.

By varying the rate at which the rectangle expands and contracts a constantly changing display is created as the rectangles interact with each other.

If the screen becomes too full, press the spacebar to clear it and start afresh.

```
10 MODE 4:VDU23,1,0;0;0;
0;;GCOL 3,1:VDU29,640;512;
20 XX=0:YX=0
30 REPEAT
40 MOVE XX,YX:DRAW XX,-Y
X:DRAW -XX,-YX:DRAW -XX,YX:
DRAW XX,YX
50 IF XX=0 AX=32*RND(2)
60 IF YX=0 BX=16*RND(2)
70 IF XX>638 AX=-AX
80 IF YX>510 BX=-BX
90 XX=XX+AX:YX=YX+BX
100 UNTIL NOT INKEY0:RUN
```

TOP 10
B-B-C-S-O-F-T-W-A-R-E

THIS MONTH	LAST MONTH	TITLE (Software House)	PRICE
1	●	**PAPERBOY** *Elite*	9.95
2	▽ 1	**SUPERIOR COLLECTION VOL. 3** *Superior*	9.95
3	▽ 2	**FOUR GREAT GAMES** *Micro Value*	3.99
4	△ 5	**FIVE STAR GAMES** *Beau Jolly*	9.95
5	▽ 4	**FOOTBALL MANAGER** *Addictive*	9.95
6	▽ 3	**DUNJUNZ** *Bugbyte*	2.99
7	●	**AROUND WORLD IN 40 SCREENS** *Superior*	6.95
8	▽ 8	**MICROPOWER MAGIC 2** *Micropower*	7.95
9	△ 10	**RAVENSKULL** *Superior*	9.95
10	●	**MICROPOWER MAGIC** *Micropower*	7.95

Compiled by Gallup/Microscope

Micropower Magic re-enters the chart this month at number 10, while Paperboy shoots to number one. Compilations are well represented as usual. Watch out for Around the World in 40 Screens which enters at number seven and is reviewed this issue.

ONE of the most useful utilities published in *Electron User* last year was Ian Brown's View Printer Driver.

This enabled you to print out text from View in a variety of type styles using the highlight codes Func+H and Func+J. Unfortunately, it did have its limitations.

Only eight highlights were allowed and the driver assumed that they were all escape sequences. This meant that on some printers a few of the type styles were unavailable.

The driver presented here isn't merely an upgrade, it's a complete rewrite. The code is extremely compact and faster to boot. This has made it possible to extend the number and range of highlights available and make it more user friendly at the same time.

You can now define up to 16 different highlights and they can be either normal, escape, or a mixture of normal and escape sequences, even in the same highlight.

This last option is essential because some type styles are turned on with an escape, but turned off with a normal sequence and vice versa.

Figure I shows 13 type styles printed on an Epson FX-80 and provides an example of what can be achieved using Driver II. Figure II shows the View editing screen with the highlights in place.

To define a highlight first move on to a blank line then press Func+O and type HT followed by Return.

Now you can enter the highlight number, one or two (I've used one throughout in Figure II), followed by the code it is to produce.

Notice that the highlight has a whole line to itself and you should not place any text on it. This blank line is not sent to the printer and you won't see a blank line on the paper.

To use the highlight press Func+H for highlight one and Func+J for highlight two. Each highlight is a simple toggle switch – the

And now – View printer overdriver

JULIE BOSWELL upgrades and extends our popular View printer driver

first time a particular highlight is met View switches the style on, the second time off, then on, off and so on.

Note that Driver II is a printer driver generator so don't try to load it into View. You must type this listing into Basic and run it.

You'll first be asked to input the pad character. This is used to prevent View from justifying or word wrapping two or more words – spacing them out or splitting them over two lines.

The pound sign is probably best for this as it is rarely used (if you want a pound you can select the UK character set on the printer, using a highlight of course, and print a hash).

On screen you'll see a pound, however, on paper it will be replaced by a space.

You'll then be asked if you wish to enter highlight number 128 – the first highlight. Press Y and you'll be prompted for the codes to turn the type style on.

At this point you may find that your printer manual says something like "ESC S 0 enables superscript mode". You can convert this to numbers and enter it as:

27,83,48

ESC is Ascii 27, S is Ascii 83 and 0 is Ascii 48. There is however, an easier way, simply enter:

ESC,ASC"S",ASC"0"

After entering the codes to switch the style on, you'll be asked for the sequence to switch it off. Enter the codes using either of the two methods or a mixture of both.

Figure I: Some of the type styles available with Driver II

Figure II: The View editing screen

◄ *From Page 41*

Note that the codes can be a normal Ascii code, escape sequence or a mixture of both and can be of different length.

After entering the first highlight you'll be prompted for the second, then the third and so on until you answer N to the prompt. The program will now create and save the machine code printer driver file under the name *Driver*. Save the Basic listing under a different name.

To use it enter View and load the printer driver with:

```
LOAD DRIVER
```

and you're ready to enter and print text.

This upgrade will probably be the most useful addition to your collection of utilities and extends the range and power of the your Electron as a word processor. ∎

```
10 a$="Printer Driver 1"
20 REM By Julie Boswell
30 REM (c) Electron User
40 MODE 6
50 ESC=27
60 PRINT''a$
70 INPUT"Input pad chara
cter:pad$
80 areg=&90:yreg=&91
90 FOR pass=4 TO 6 STEP 2
100 P%=&400:O%=&C00
110 [OPT pass
120 JMP print.char
130 JMP printer.on
140 JMP printer.off
150 JMP exit
160 JMP exit
170
180 .on EQUB 0
190
200 .prins
210 PHA
220 LDA #1:JSR &FFEE
230 PLA
240 AND #&7F:JMP &FFEE
250
260 .printer.off
270 LDA #0:STA on
280 LDA #3:JMP &FFEE
290
300 .printer.on
310 LDA on:BNE exit
320 INC on
330 LDA #2:JSR &FFEE
340 LDA #ESC:JSR print
350 LDA #ASC"@":JSR print
360 .exit
370 RTS
380
390 .print.char
400 CMP #ASC pad$:BNE not.
pad
410 LDA #32
420 .not_pad
430 CMP #128:BPL highlight
440 JMP &FFE3
450 .highlight
460 CMP #&44:BCS exit
470 STA areg:STY yreg
480 AND #&7F:ASL A:ASL A:T
AX
490 LDA table+1,Y:EOR #&80
:STA table+1,Y
500 AND #&80:BNE here
510 INY:INY
520 .here
530 LDA table,Y:AND #&30:B
EQ no.esc
540 LDA #27:JSR print
550 .no_esc
560 LDA table,Y:JSR print
570 LDA table+1,Y:JSR prin
t
580 LDA areg:LDY yreg
590 RTS
600
610 .table
620 ]
630 NEXT
640 i=128
650 PRINT"Enter highlight
";i;"(Y/N)?";
660 IF INSTR(Nn,GET$) DS
CLI"SAVE Driver C00 D00":END
670 INPUT LINE"On:a$
680 INPUT LINE"Off:b$
690 !O%=FNcode(a$)+&10000*
FNcode(b$)
700 O%=O%+4::i=i+1
710 GOTO 650
720
730 DEF FNcode(a$)
740 C%=0
750 I%=INSTR(a$,",")
760 C%=C% OR EVAL(a$)
770 IF I%=0 THEN =C%
780 a$=MID$(a$,i+1)
790 IF C%<>27 THEN =C%&10
0&EVAL(a$)
800 C%=&80:GOTO 250
```

> This listing is included in this month's cassette tape offer. See order form on Page 53.

·ARENA·

By MICHAEL JONES

THE Arena is an arcade adventure for two players, in which, as a denizen of Ancient Rome, you go hunting for treasure against immeasurable odds.

In the Arena, two players fight and each controls his character inside its own scrolling window, shown in Figure I.

The object is to collect treasure chests scattered throughout a large maze. Each player needs to collect eight. Player one collects chests marked 1 and player two, chests marked 2.

When they have all been collected you have to reach the finish before your oxygen runs out. This is can be found at the tip of the large arrow that forms part of the wall at the bottom centre of the playing area. This area can only be reached by a teleport.

Oxygen levels become depleted faster by running or bumping into walls. More oxygen can be picked up by running over the ox symbols.

Figure II shows a full list of the control keys and objects to be found.

The skulls and protectors are fatal to the touch. The protectors may be destroyed using a neutraliser, skulls on the other hand must always be avoided.
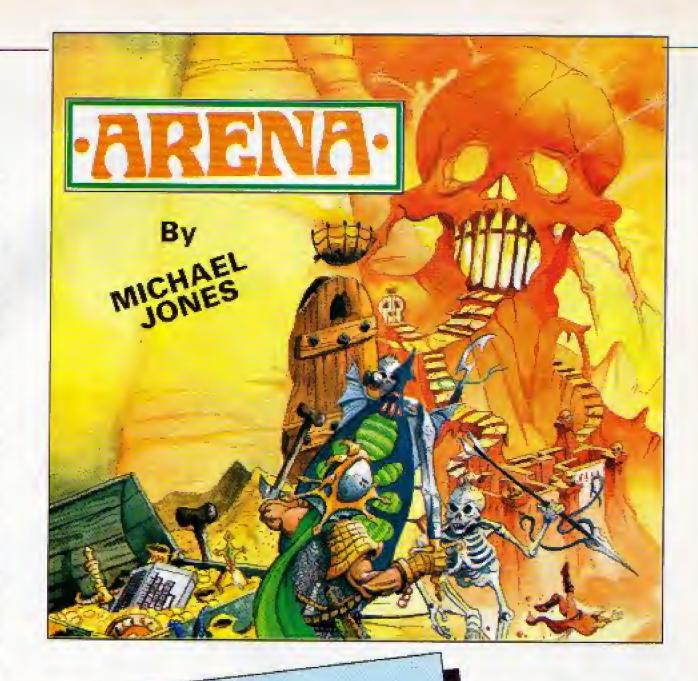
Dropping a sandbag leaves a pile of sand, which may trap the other player since they can only be removed with a spade.

To pick up an object simply walk over it. Unfortunately, your shield is heavy, and you can only carry one object at a time, although objects that are used immediately, treasure chests for instance, can be picked up any time.

The game ends when a player finishes or both players die, the winner in this case being the one with the highest score.

Although the Arena is intended for two players it can be played by one. This can be an advantage if you want to practice and beat your friends to the treasure.

Due to the compact design of the program it should be entered with care

and saved before being run.

Be careful only to enter spaces where they are printed, except the ones between the line number and the start of the line. For instance enter line 650 as:

```
650IFINKEY-67ANDADIRX=0ADI
RX=4:NAX=AX-1:mX=mX+1ELSEI
FINKEY-83ANDADIRX=0ADIRX=2
:NAX=AX+1:mX=mX-1
```

without any spaces, and not as:

```
650 IF INKEY(-67) AND ADIR
X=0 ADIRX=4:NAX=AX-1:mX=mX
+1 ELSE IF INKEY-83 AND AD
IRX=0 ADIRX=2:NAX=AX+1:mX=
mX-1
```
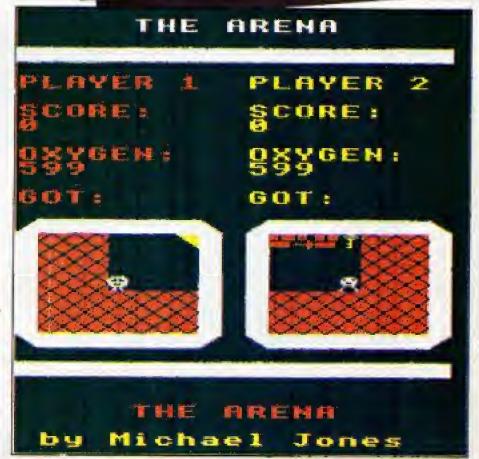
However, some spaces are essential and have been left in certain lines.
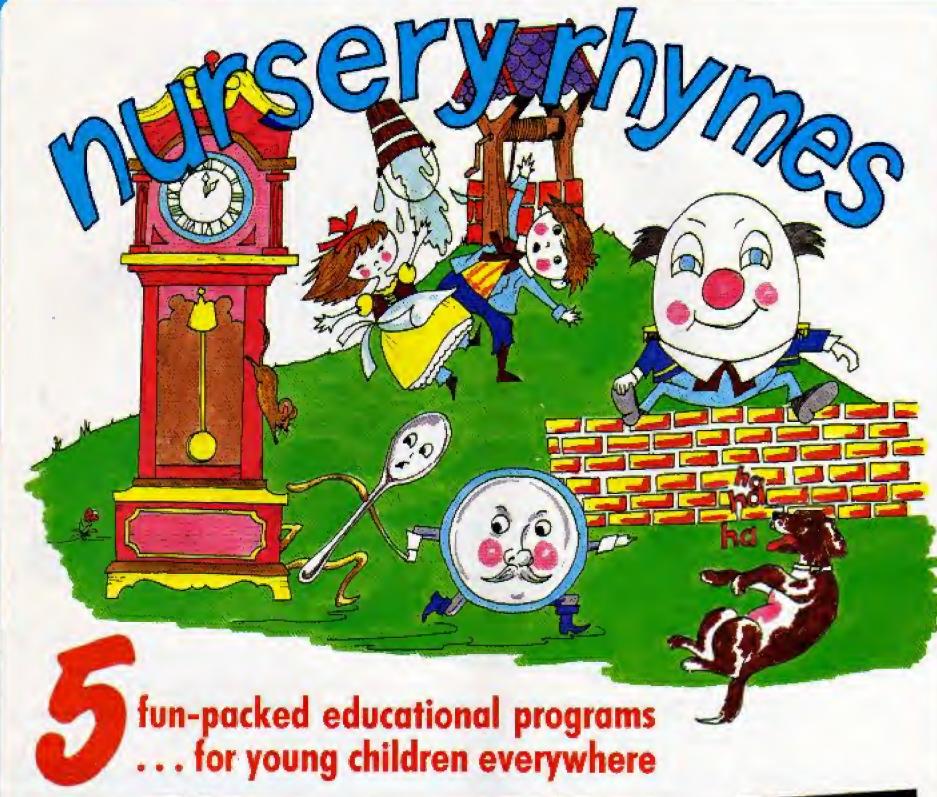
Watch out for variables with similar names, like O2% and O2. Also S and 5 look very similar, for example:

Enter the Arena at your peril, your destiny is upon you.

Full listing starts on Page 45

Figure I: The main playing screen

```
   10 REM The Arena
   20 REM by Michael Jones
   30 REM (c) Electron User
   35 IF PAGE>&E00 GOTO 1700
   40 MODE6:VDU23,1,0;0;0;0;
:HIMEM=&C740:PROCassemble:PR
OCinitdata:VDU22,5:PROCchars
:REPEAT:PROCtitle:CALLprodat
:PROCinit:PROCscr:REPEAT:IFD
1%=FALSE PROCplayer1:IFD1%PR
OCdie(4,A%)
   50 IFD2%=FALSE PROCplayer
2:IFD2%PROCdie(14,D%)
   60 UNTILW1%ORW2%ORLD1%AND
D2%)
   70 IFD1%ANDD2%PROCbothlos
e ELSEPROCwin.
   80 REPEATUNTILGET$=" ":UN
TILFALSE
   90 END
  100 DEFPROCinitdata:Z%=&47
40:CLS:PRINTTAB(14,2)"Please
 wait";:FORloop=1TO27:READd
$:FORloop1=1TO6:IFMID$(d$,l
oop1,1)="/"Z%=32ELSE?Z%=ASC
(MID$(d$,loop1,1))+159
  110 Z%=Z%+1:NEXT:FORloop2=
1TO3:?Z%=232:Z%=Z%+1:NEXT:;F
ORloop3=1TO6:FORloop4=1TO64:
IFloop4<62ANDloop3<4?Z%=32EL
SE?Z%=232
  120 Z%=Z%+1:NEXT:ENDPROC
  130 DATA"FFFFFF/BR//DEDEDE
DE//DEH//H//////G//////////
/CE///GAELLEUE",IS///F/AB
//EDEDEDE/QDES//R/H/HHH/G/_
_/_____////CAAE/E//EEE"/
/////F/AE//DEDEDE//DEHH/HH///
/NH/G/__/_/_II_//////DB//CE//EE
///K"
  140 DATA"/W///M/E///EDEDE/
/DE/H///HHH/HHH/G2//X_/U.///.
/DAAB////DEEEEEE",FFFFFF//GG
G/GGGGGG/GR///H//H/H//////GGGG
GGGGGGG/GGGGGGGGGGGGGGGG","V
HHH///////////G/GGGGGG//DAA
AAE//////I///GGG///AAAAE////
CAAA"
  150 DATA"/HRH/IIIIIIIII//
//////DEDEDE///IIIIIII/G//
/G/AAAE//DB//CAA","/H/H/I///
///HHHHHHHHHHHDEDEDE//_/I//
////I/G/GGG/AAE//DECB//CA"/
H////I/FFFFF/HH//////HHHH/////
///./I/II/I/G//G/AE//DET/C
B//C"
  160 DATA"/HH//I/FMMMM//H/H
HH////H///DAAE//_//I//I//I/G
///A//DAAB//CB//","/HH/I/FM
UMF/H/RH//HHHH///DE//./I/I
/I/I/G/G/GGA/DAN/AB//CB/","B
//H//I/FMMME//HHHH/HH/VHHDAE/
0//.I//ITI/I/G//G//A/ACA/EC/
AVAM"
  170 DATA"C9//H//ISFFFFF/HH//
//HH/HHDDDDDEEEEEEI/III/I/GGG
G/GA/ARA/DB/A/A/","PA///IHHH
HHH//HH/HHHH//////////FQF////I/
II//I/GG//GA/A/A/EC/A/A/","/
A/B////////QH//H//HT/M/HHH/FE/E
////F/I/EII/I/G//G/GA/A/A/DB/
A/A/"
  180 DATA"/A/AAAA//T/HH/H/H
HHH///HUF//////F/F//////I/G/G
GGGA/A/A/EC/A/A/",////AAAAB/
//HH/H//////HHHH/HFFFFFFFFM/II
```

```
MIII/G///GGA/A/A/DB/A/A/","A
AAAAAA/I////H/HH//H/H////
////////////G/G//////////EC/
////K"
  190 DATA"AE///CAAG/GGGGH//
H/H//H/GGGGGGGGGGGGGGGGGGG/G
GGGGGGGGGGGGGGG",A//D//CAB
////HHJHHHHHHH///G/_/////G
/////////////////N",A
/CAB//ABCAB///////QGG//G/
/GG/G//G/GGGAA/BDGDEGGGCACAA
AAAA"
  200 DATA"A/M/CAB/AABCAFFFF
FF/FFFF/G/G./G//G//G/GN//A/A
A/AB/PG//////////",ADE///A//
ECBC/IT//E/E.///G/GGGGDBGGGG
/G///E/EC//CB//G//////////","C
A//./AB///CB//Q//F/E//_UG///
/DAAB///G/////////////G/N//_
```

```
  210 DATA"SCB///CAAB//CB///
/J/F/_/DB/Q///AA///L/DB/AB/D
E/AB/DB////_T///."//CB////U
AB//CB////F/E///AA///_AAO///
/ECVAE/AB/ACDEC///QFFFFM",/
/CAAAAAECB/ZAAAAAA/AAAAAAAA
ABAADAAA/ECGABGCBGEDGECDAABF
FYFG"
  220 DEFPROCchars:VDU23,1,0
;0;0;0;19,1;6,0,0,0,19,2,2,0
,0,0,0,23,224,255,255,255,255,
255,255,255,255,23,225,128,1
92,224,240,248,252,254,255,2
3,226,255,127,63,31,15,7,3,1
,23,227,1,3,7,15,31,63,127,2
55,23,228,255,254,252,248,24
0,224,192,128
  230 VDU23,229,239,239,239,
0,254,254,254,0,23,230,204,5
1,204,51,204,51,204,51,23,23
1,126,195,129,129,129,129,19
5,126,23,232,231,219,189,126
,126,189,219,231,23,233,0,8,
24,20,60,46,125,223,23,234,1
24,214,146,254,198,124,40,10
8
  240 VDU23,235,60,90,126,10
2,60,36,66,195,23,236,255,85
,255,85,255,85,255,85,23,237
,0,124,195,238,238,238,124,0
,23,238,0,60,66,90,90,66,60,
0,23,239,56,40,16,16,16,124,
124,56,23,240,28,34,28,8,8,2
4,8,24,23,241,60,24,60,126
,255,255,126
  250 VDU23,242,247,187,87,2
38,213,187,223,239,23,243,12
6,255,0,231,247,247,247,247,
23,244,126,255,0,195,255,195
,223,195,23,245,0,0,234,164,
234,0,0,0,23,246,112,80,112,
4,6,255,6,4,23,247,14,10,14,
32,96,255,96,32
  260 VDU23,248,184,168,184,
4,6,255,6,4,23,249,23,21,23,
32,96,255,96,32,23,250,0,60,
126,126,126,126,60,0,23,251,
0,48,112,112,126,126,60,0,23
,252,0,0,0,0,126,126,60,0,23
,253,0,0,0,0,14,14,12,0,23,2
54,56,124,84,124,124,56,56,0
  270 VDU23,255,16,34,23,136
,84,34,4,8:ENDPROC
  280 DEFPROCtitle:CLS:COLOU
R1:s$=CHR$13+CHR$10+CHR$9+CH
R$9+CHR$9:t$=s$+CHR$9:PRINTT
AB(5,2)"THE ARENA";TAB(3,7)
CHR$244;" SCORE RESET";t$;C
```


Figure II: The controls

```
HR$248;" BONUS";s$;CHR$245;"
OXYGEN";t$;:COLOUR2
  290 PRINTCHR$238;" JEWEL";
s$;CHR$239;" SPADE";t$;CHRS
240;" KEY";s$;CHR$241;" SAN
DBAG";t$;CHR$242;" NEUTRALIZ
ER";s$;CHR$243;" CHEST";TAB
(0,30)"Press Space to Start"
;:COLOUR3
  300 PRINTTAB(4,16)CHR$233;
" SAND PILE";s$;CHR$236;" G
RILLE";t$;CHR$237;" TELEPORT
";s$;CHR$254;" SKULL";t$;CH
R$255;" PROTECTOR";TAB(2,4)"
by Michael Jones";TAB(0,22)"
CONTROLS: 1 2";:COLOUR1
  310 PRINTTAB(0,23)"DROP LE
FT Ctrl M DROP RIGHT V
 Ret MOVE LEFT X , MOV
E RIGHT C . MOVE UP
 A ; MOVE DOWN Z /
":REPEATUNTILGET$=" ":ENDPROC
  320 DEFPROCinit:W1%=FALSE:
W2%=FALSE:A%=&560:B%=&56FC:
ADIRX=0:BDIRX=0:C1%=0:C2%=0:
S1%=0:S2%=0:O1%=600:O2%=600:O1
%=01:O2%=02:CHE1%=0:CHE2%=0:
MX=60:N%=0:COLOUR3:CLS:REPEA
T:D1%=FALSE:D2%=FALSE
  330 PRINTTAB(0,2)"Player 1
? (Y/N)":IFGET$="N"D1%=TRUE
  340 PRINTTAB(0,4)"Player 2
? (Y/N)":IFGET$="N"D2%=TRUE
  350 CLS:UNTILD1%=FALSE ORD
2%=FALSE:ENDPROC
  360 DEFPROCscr:COLOUR3:PRI
NTTAB(5,1)"THE ARENA";TAB(0,
3)STRING$(19,CHR$224);TAB(0,
25)STRING$(19,CHR$224);TAB(
,15)STRING$(19,CHR$224);TAB(
0,23)STRING$(19,CHR$224);TAB
(0,15)CHR$227;TAB(10,15)CHR$
227;TAB(0,23)CHR$226;TAB(10,
23)CHR$226;
  370 FORloop%=16TO22:PRINTT
AB(0,loop%)CHR$224;TAB(18,lo
op%)CHR$224;TAB(10,loop%)CHR$
224;TAB(18,loop%)CHR$224;:NE
XT
  380 PRINTTAB(8,15)CHR$225;
```

```
TAB(18,15)CHR$225;TAB(8,23)C
HR$228;TAB(18,23)CHR$228;TAB
(9,15)" ";TAB(9,23)" ";:COLO
UR1:PRINTTAB(5,28)" THE ARENA
":PROCdisp(1,0):COLOUR2:PRIN
TTAB(1,30)"by Michael Jones"
:PROCdisp(2,10)
  390 PROCwindow(A%,1):PROCw
indow(B%,11):ENDPROC
  400 DEFPROCdisp(p%,x%):PRI
NTTAB(x%,5)"PLAYER ";p%;TAB(
x%,7)"SCORE:";TAB(x%,8)"0";T
AB(x%,10)"OXYGEN:";TAB(x%,11
)"600";TAB(x%,13)"0";:END
PROC
  410 DEFPROCwindow(mp%,x%):
FORloop%=0TO6:FORloop1%=0TO96
:E%=((loop1%*64)+mp%-195+loo
p%):?(z+1)=E%MOD256:?(z+2)=E
%DIV256:CALLz:PRINTTAB(x%+lo
op%,16+loop1%)CHR$(?E%);:NE
XT;:ENDPROC
  420 DEFPROCassemble:DIMAX%
40:FOR1=0TO2STEP2:P%=Q%
  430 [OPTI
  440 .scrleft LDA#7:STA&8F:
.lpa1 LDX#0:.aa LDA&4242,X:.
ab STA&4242,X:INX:CPX#96:BNE
aa:LDAaa+1:CLC:ADC#64:STAaa+
1:LDAaa+2:ADC#1:STAaa+2:LDAa
b+1:CLC:ADC#64:STAab+1:LDAab
+2:ADC#1:STAab+2:DEC&8F:BNE1
pa1
  450 SEC:LDA&80:SBC#3:STAt+
1:STAz+1:LDA&81:SBC#1:STAt+2
:STAz+2:LDX#7:.w CLC:LDAt+1:
ADC#64:STAt+1:STAz+1:LDAt+2:
ADC#0:STAt+2:STAz+2:JSRz:JSR
y:DEX:BNEw:RTS
  460 .scrright LDA#7:STA&8F
:.lpb1 LDX#96:.ba LDA&4242,X
:.bb STA&4242,X:DEX:BNEba:LD
Aba+1:CLC:ADC#64:STAba+1:LDA
ba+2:ADC#1:STAba+2:LDAbb+1:C
LC:ADC#64:STAbb+1:LDAbb+2:AD
C#1:STAbb+2:DEC&8F:BNE1sb1
  470 SEC:LDA&80:SBC#253:STA
t+1:STAz+1:LDA&81:SBC#0:STAt
```

```
+2:STAz+2:LDX#7:JMPw
  480 .scrdown SEC:LDA&80:SB
C#196:STAu+1:STAz+1:LDA&81:S
BC#0:STAu+2:STAz+2:LDX#7:.v
CLC:LDAu+1:ADC#1:STAu+1:STAz
+1:LDAu+2:ADC#0:STAu+2:STAz+
2:JSRz:JSRx:DEX:BNEv:RTS
  490 .scrup CLC:LDA&80:ADI#
188:STAu+1:STAz+1:LDA&81:ADC
#0:STAu+2:STAz+2:LDX#7:JMPv
  .z LDA&4242:SEC:SBC#229:BCCm
ag:SEC:SBC#4:BCCred:SEC:SBC#
5:BCCwhi:SEC:SBC#7:BCCmag:SE
C:SBC#5:BCCred:JMPwhi
  500 .mag LDA#17:JSR&FFEE:L
DA#2:JSR&FFEE:RTS:.red LDA#1
7:JSR&FFEE:LDA#1:JSR&FFEE:RT
S:.whi LDA#17:JSR&FFEE:LDA#3
:JSR&FFEE:RTS
  510 .y LDA#31:JSR&FFEE:LDA
&82:JSR&FFEE:LDA&83:JSR&FFEE
:.x LDA&4242:JSR&FFEE:INC&83
:RTS:.x LDA#31:JSR&FFEE:LDA&
82:JSR&FFEE:LDA&83:JSR&FFEE:
.u LDA&4242:JSR&FFEE:INC&82:
RTS
  520 .prodat LDA#&47:STAca+
2:LDA#&4F:STAcb+2:LDA#&4D:ST
Aca+1:LDA#&80:STAcb+1:LDA#9:
STA&8F:LDX#0:.pc1:.ca LDA&4
740,X:.cb STA&4F80,X:INX:BNE
lpc1:INCca+2:INCcb+2:DEC&8F:
BNElpc1:RTS
  530 .]
  540 NEXT:ENDPROC
  550 DEFPROCscroll1:IFADIR%
=2 ?(aa+1)=&20:?(aa+2)=&6C:?
(ab+1)=&10:?(ab+2)=&6C:?&82=
7:?&83=16:?&80=(A%+6)MOD256:
?&81=(A%+6)DIV256:CALLscrlef
t
  560 IFADIR%=4 ?(ba+1)=&0F:
?(ba+2)=&6C:?(bb+1)=&1F:?(bb
+2)=&6C:?&82=1:?&83=16:?&80=
(A%-6)MOD256:?&81=(A%-6)DIV2
56:CALLscrright
  570 IFADIR%=1VDU28,1,22,7,
16:PRINTTAB(0,0)CHR$11:CHR$2
6:?&82=1:?&83=16:?&80=A%MOD2
56:?&81=A%DIV256:CALLscrdown
  ELSEIFADIR%=3VDU28,1,22,7,
16:PRINTTAB(6,6):VDU26:?&82=1
6:?&83=22:?&80=A%MOD256:?&81=
A%DIV256:CALLscrup
  580 ENDPROC
  590 DEFPROCscroll2:IFBDIR%
=2 ?(aa+1)=&20:?(aa+2)=&6C:?
(ab+1)=&10:?(ab+2)=&6C:?&82=
17:?&83=16:?&80=(B%+6)MOD256
:?&81=(B%+6)DIV256:CALLscrle
ft
  600 IFBDIR%=4 ?(ba+1)=&AF:
?(ba+2)=&6C:?(bb+1)=&BF:?(bb
+2)=&6C:?&82=1:?&83=16:?&80
=(B%-6)MOD256:?&81=(B%-6)DIV
256:CALLscrright
  610 IFBDIR%=1VDU28,11,22,1
7,16:PRINTTAB(0,0)CHR$11:CHR
$26:?&82=11:?&83=16:?&80=B%M
OD256:?&81=B%DIV256:CALLscrd
own ELSEIFBDIR%=3VDU28,11,22
,17,16:PRINTTAB(6,6):VDU26:?
&82=11:?&83=22:?&80=B%MOD256
:?&81=B%DIV256:CALLscrup
  620 ENDPROC
  630 DEFPROCplayer1:M%=M%:n
%=N%:ADIR%=0:NA%=A%:nA%=0:IF
```

```
INKEY-66ADIR%=1:NA%=A%+64:n%
=n%+1 ELSEIFINKEY-98ADIR%=3:
NA%=A%+64:n%=n%-1
  640 IFINKEY-67ANDADIR%=0AD
IR%=4:NA%=A%-1:m%=m%+1 ELSEI
FINKEY-83ANDADIR%=0ADIR%=2:N
A%=A%+1:m%=m%-1
  650 IFADIR%=0AND(1%<>0ANDI
NKEY-2PROCdrop(1,A%-1,-1) EL
SEIFADIR%=0AND(1%<>0ANDINKEY
-100PROCdrop(1,A%+1,1)
  660 CODE%=?NA%:IFCODE%=32A
NDADIR%>0AM%=1:GOTO700 ELSEI
FCODE%=233AND(1%=239PROCsand
(NA%,1):AM%=1:GOTO700
  670 IFCODE%=255AND(1%=242P
ROCneut(NA%,1):AM%=1:GOTO700
  ELSEIFCODE%=238ORCODE%=243O
R(CODE%>244ANDCODE%<250)PROC
inget(NA%,1):AM%=1:GOTO700
  680 IFCODE%=238ANDCODE%<24
3AND(1%=0PROCget(NA%,1):AM%=
1:GOTO700 ELSEIFCODE%=236AND
(1%=240PROCgrille(NA%,1):AM%
=1:GOTO700
```

```
  690 IFCODE%=237PROCtelepor
t(NA%,1) ELSEIFCODE%=254ORCO
DE%=255D1%=TRUE:AM%=1
  700 IFO1%=0P1%=TRUE
  710 IFADIR%=0AND001%>001=01
-0.2 ELSEO1=O1-1
  720 O1%=O1:IFAM%=1?A%=32:P
RINTTAB(4,19)" ":A%=NA%:?A%
=0:COLOUR3:IFABS(M%)<4ANDABS
(N%)<4AND02%=FALSE PRINTTAB(
4+M%,19+N%)" ";TAB(14-M%,19-
N%)" ";
  730 IFAM%=1PROC(scroll1:COL
OUR3:IFABS(m%)<4ANDABS(n%)<4
AND02%=FALSE PRINTTAB(4+m%,1
9+n%)CHR$235;TAB(14-m%,19-n%
)CHR$234;
  740 COLOUR3:PRINTTAB(4,19)
CHR$235;:IF(HE1%=8ANDA%=&54D
F S1%=S1%+50:W1%=TRUE
  750 M%=(B%MOD64)-(A%MOD64)
:N%=(B%DIV64)-(A%DIV64)
  760 IFNOTO1%COLOUR1:PRINTT
AB(0,11)O1%;" ";TAB(0,8);S1
%;" ";
  770 ENDPROC
  780 DEFPROCplayer2:m%=M%:n
%=N%:BDIR%=0:NB%=B%:BM1=0:IF
INKEY-73BDIR%=1:NB%=B%-64:n%
=n%-1 ELSEIFINKEY-105BDIR%=3
:NB%=B%+64:n%=n%+1
  790 IFINKEY-103ANDBDIR%=0B
DIR%=4:NB%=B%-1:m%=m%-1 ELSE
IFINKEY-104ANDBDIR%=0BDIR%=2
:NB%=B%+1:m%=m%+1
  800 IFBDIR%=0AND(C2%<>0ANDI
NKEY-102PROCdrop(2,B%-1,-1)
ELSEIFBDIR%=0ANDC2%<>0ANDINK
EY-74PROCdrop(2,B%+1,1)
  810 CODE%=?NB%:IFCODE%=32A
NDBDIR%>0BM%=1:GOTO850 ELSEI
FCODE%=233AND(C2%=239PROCsand
(NB%,2):BM%=1:GOTO850
  820 IFCODE%=255AND(C2%=242P
ROCneut(NB%,2):BM%=1:GOTO850
```

```
  ELSEIFCODE%=238OR(CODE%>243
ANDCODE%<250)PROCinget(NB%,2
):BM%=1:GOTO850
  830 IFCODE%>238ANDCODE%<24
3AND(2%=0PROCget(NB%,2):BM%=
1:GOTO850 ELSEIFCODE%=236AND
C2%=240PROCgrille(NB%,2):BM%
=1:GOTO850
  840 IFCODE%=237PROCtelepor
t(NB%,2) ELSEIFCODE%=254ORCO
DE%=255S2%=TRUE:BM%=1
  850 IFO2%=0D2%=TRUE
  860 IFBDIR%=0AND02%>002=02
-0.2 ELSEO2=O2-1
  870 O2%=O2:IFBM%=11B%=32:P
RINTTAB(14,19)" ";:B%=NB%:?B
%=0:COLOUR3:IFABS(M%)<4ANDAB
S(N%)<4AND02%=FALSE PRINTTAB
(4+M%,19+N%)" ";TAB(14-M%,19
-N%)" ";
  880 IFBM%=1PROCscroll2:COL
OUR3:IFABS(m%)<4ANDABS(n%)<4
AND01%=FALSE PRINTTAB(4+m%,1
9+n%)CHR$235;TAB(14-m%,19-n%
)CHR$234;
  890 COLOUR3:PRINTTAB(14,19
)CHR$235;:IF(HE2%=8ANDB%=&54
DF S2%=S2%+50:W2%=TRUE
  900 M%=(B%MOD64)-(A%MOD64)
:N%=(B%DIV64)-(A%DIV64)
  910 IFNOTO2%COLOUR2:PRINTT
AB(10,11);O2%;" ";TAB(10,8);
S2%;" ";
  920 ENDPROC
  930 DEFPROCbothlose:CLS:MO
VE320,960:VDU5:GCOL48,3:PRIN
T'GAME OVER':VDU4:COLOUR2:P
RINTTAB(0,5)'Player 1 score:
';S1%;:COLOUR1:PRINTTAB(0,9)
'Player 2 score:';S2%;:COLOU
R3:PRINTTAB(0,20)'Press Spac
e to Start';:ENDPROC
  940 DEFPROCwin:FORS0%=100T
O250STEP10:SOUND1,-15,S0%,1:
NEXT:CLS:MOVE320,960:VDU5:GC
OL48,3:PRINT'GAME OVER':VDU
4:COLOUR2:PRINTTAB(0,5)'Miss
ion completed by';SPC6,'play
er ';:IFW1%PRINT'1'ELSEPRINT
'2'
  950 COLOUR3:PRINTTAB(0,9)'
Player 1 score:';S1%;:COLOU
R1:PRINTTAB(0,13)'Player 2 s
core:';S2%;:COLOUR2:PRINTTA
B(0,20)'Press Space to Start
';:ENDPROC
  960 DEFPROCdie(x%,p%):COLO
UR3:FORloop%=1TO4:PRINTTAB(x
%,19)CHR$(249+loop%):SOUND&1
1,-15,45+(10*loop%),3:FORloo
p2%=1TO150:NEXT:,PRINTTAB(x%
,19)" ";:?p%=32:FORS0%=255TO
0STEP-1:SOUND&11,-15,S0%,1:S
OUND&10,-15,5,1:NEXT:ENDPROC
  970 DEFPROCdrop(p%,x%,C%):
IF?x%<>32ENDPROC ELSEIF(p%=1
AND(1%=241)OR(p%=2ANDC2%=241
)?x%=233
  980 PRINTTAB((p%*10)-6,13)
" ";:SOUND1,-15,100,2:SOUND1
,-15,80,2:SOUND1,-15,150,2:1
Fp%=1AND(1%<>241?xX%=C1%ELSE1
Fp%=2AND(2%<>241?xX%=C2%
  990 IFp%=1C1%=0ELSEC2%=0
  1000 COLOUR2:IF?xX%=233COLOU
R3
  1010 IFp%=2IFABS(MX+L%)<4AN
DABS(N%)<4PRINTTAB(4+L%+M%,1
9+N%);CHR$(?x%);
```

```
  1020 IFpX%=1IFABS(M%-L%)<4AN
DABS(N%)<4PRINTTAB(14+L%-M%,
19-N%);CHR$(?x%);
  1030 PRINTTAB((pX%*10)-6+L%,
19);CHR$(?x%);:ENDPROC
  1040 DEFPROCget(x%,p%):IFp%
=1C1%=CODE%ELSEC2%=CODE%
  1050 SOUND1,-15,100,2:SOUND
1,-15,150,2:COLOUR3:PRINTTAB
((p%*10)-6,13)CHR$CODE%;:END
PROC
  1060 DEFPROCneut(x%,p%):?x%
=32:IFp%=1C1%=0:S1%=S1%+10EL
SEIFp%=2C2%=0:S2%=S2%+10
  1070 PRINTTAB((p%*10)-6,13)
" ";:SOUND1,-15,5,10:ENDPROC
  1080 DEFPROCinget(x%,p%):IF
CODE%=245ANDp%1COLOUR1:01=0
1+500:C1%=01:PRINTTAB(0,11)
01%;ELSEIFCODE%=245ANDp%=2CO
LOUR2:02=02+500:02%=02:PRINT
TAB(10,11);02%;
  1090 IFCODE%=246S2%=0ELSEIF
CODE%=247S1%=0
  1100 IFCODE%=248S2%=S2%-10E
LSEIFCODE%=249S1%=S1%+10
  1110 IFCODE%=238ANDp%=1S1%=
S1%+20ELSEIFCODE%=238ANDp%=2
S2%=S2%+20
  1120 COLOUR2:IFCODE%=243CHE
1%=CHE1%+1:SOUND1,-15,10,2:S
1%=S1%+50:PRINTTAB(0,5)STRIN
G$(CHE1%,CHR$243)ELSEIFCODE%
=244CHE2%=CHE2%+1:SOUND1,-15
,10,2:S2%=S2%+50:PRINTTAB(19
-CHE2%,3)STRING$(CHE2%,CHR$2
44)
  1130 SOUND1,-15,50,4:SOUND1
,-15,250,1:COLOUR1:PRINTTAB(
0,8);S1%;"   ":COLOUR2:PRIN
TTAB(10,8);S2%;"   ":ENDPRO
C
  1140 DEFPROCgrille(x%,p%):I
Fp%=1C1%=0:S1%=S1%+15ELSEC2%
=0:S2%=S2%+15
  1150 PRINTTAB((10*p%)-6,13)
" ";:SOUND1,-15,50,2:SOUND1,
-15,90,2:SOUND1,-15,130,2:EN
DPROC
  1160 DEFPROCsand(x%,p%):IFp
%=1S1%=S1%-10ELSES2%=S2%-10
  1170 IFS1%<1S1%=0ELSEIFS2%<
1S2%=0
  1180 SOUND1,-15,5,4:SOUND1,
-15,7,4:ENDPROC
  1190 DEFPROCteleport(x%,p%)
:IFx%=&547Cn%=&54E2 ELSEIFxX%
=&54E7n%=&54A3
  1200 IFx%=&55?5n%=&545C ELS
EIFx%=&5234n%=&540D
  1210 IFp%=1A%=n%:PROCwindow
(A%,1):S1%=S1%+30 ELSEB%=n%:
PROCwindow(B%,11):S2%=S2%+30
  1220 M%=(B%MOD64)-(A%MOD64)
:N%=(B%DIV64)-(A%DIV64):FORS
0%=10TO100STEP10:SOUND1,-15,
S0%,1:NEXT:ENDPROC
  1700 *K.0+7.M%=PAGE-&E00:
FORI%=PAGE TO TOP STEP4:!(I%
-0%)=!I%:NEXT:PA.=&E00:MOLD:
MRUN:M
  1720 *FX138,0,128
```

Micro Messages

Computerised male chauvinists

Glancing through Electron User in which my husband's nose is perpetually buried I noted that many of the letters praised the excellence of the contents. I also noticed that most of the subscribers were male.

Have those responsible for this magazine ever considered the side effects of this so-called excellence — to wit, the effect on the wives of the poor besotted Electron addicts?

When my husband brought this infernal machine home some years ago a strange metamorphosis took place.

What used to be a typically normal man — you know the sort of thing, seen around the house regularly, chatting, grumbling, joking, complaining and so on — turned into a semi-recluse, closeted with his Electron and tortured thoughts until the small hours and appearing in the mornings looking haggard and thoughtful.

There was a ray of hope about six months later when one morning he declared peevishly: "This little Electron is no use. It's just a toy, really".

Celebration all round. He has kicked the habit. He is back with us in the real world of domestic normality.

Alas it did not last. One fateful day he picked up this magazine Electron User. The addiction returned, with sinister variations.

Only half his spare time is now spent with the perfidious contraption. The other half is spent boring everyone with mumbo-jumbo about assemblers, recursion, spare bits, ULA called Sheila (this really worries me) and the adulation of some space age deity called Waddy Love.

There seems to be no cure for this malaise. The only consolation is that it could be worse (drink, women, gambling, fishing or Elvis Presley). And at least it keeps him at home, sort of.

However, the addiction might be tempered if the publishers of this magazine would tone down this excellence and make it a bit more boring.

I feel somehow that this appeal will fall on computerised male chauvinistic ears, but it should at least give them a conscience. And let the other computer widows know that they are not alone. — **Mrs Helen Horgan, Didcot, Oxon.**

● If you can't beat 'em, join 'em — try our knitting software. You'll find an order form on Page 61.

Strongman help wanted

I WOULD like to congratulate you on producing such an excellent magazine. It has everything that you need to know about the Electron and has a good range of software.

One problem I have is finding some of the well known board games. Can you get any of the popular games?

I also need help with 5 Star Games — I can't seem to work Geoff Capes Strongman. Can someone please tell me how? —**T. Rouse, Godalming, Surrey.**

● We haven't heard of Electron versions of the well known board games apart from Chess, Draughts and Reversi — all available from Acornsoft. Can any readers help with Geoff Capes hints and tips?

Not enough room?

I FIND your publication very useful and entertaining. I have, however, nearly lost my mind trying to get Mandelbrot (Electron User May 1987) running.

I have triple checked the listing and the machine code assembles OK, but when I try to run it, it says "No room at line 150".

I have tried altering lines 150 and 440 to various positions in memory but to no avail. Can you please help me before they come to take me away!

Also any ideas on differentiation on the Electron would be appreciated. —**G. Crawford, Rainham, Kent.**

● It sounds like you have made an error in the assembly language section. It's impossible to tell where, so the only solution is to check through the listing line by line.

The machine code Mandelbrot set was a direct conversion of a simple Basic listing. This may help you follow the machine code through:

```
10 REM Mandelbrot Set
20 MODE 5
30 VDU23,1,0;0;0;0;
40 VDU29,1156;512;
50 FOR i=-1.8 TO 0.8 STEP 2/160
60 FOR j=-1 TO 1 STEP 2/256
70 GCOL 0,3
80 PLOT 69,i*640,j*512
90 NX=-1:x=0:y=0
100 REPEAT
110 x2=x*x:y2=y*y
120 y=2*x*y+j:x=x2-y2+i
130 NX=NX+1
140 UNTIL NX=50 OR x2+y2>=4
150 IF NX<5 GCOL 0,3
160 IF NX>4 AND NX<16 GCOL 0,2
170 IF NX>15 AND NX<49 GCOL 0,1
180 IF NX=50 GCOL 0,0
190 PLOT 65,0,0
200 NEXT
210 NEXT
```

Can any readers help Mr Crawford with his differentiation?

1st class service

MAY I as a relative newcomer to Electron computing, express my thanks and gratitude for the excellent backup provided by Slogger. I visited the

Tails in the air

LAST month I bought an Acorn-Olivetti JP101 printer by mail order on special offer at £67.85. Although limited, it is quite a useful printer.

However, I was surprised to find that lower case letters rest slightly, but noticeably higher than upper case letters. Where there is a mixture of both it gives lines of text a misaligned appearance. For Olivetti, with long experience in typesetting, this seems to be most unusual.

I wrote to the suppliers of the printer and to the magazine carrying their advertisment, but haven't received a reply. My question is whether or not this misalignment was intended or is a defect. —**S. Braithwaite, Lincoln.**

● A lack of proper descenders is one of the features of an inexpensive printer.

You will only get true descenders on a printer costing several hundred pounds.

Electron & BBC Micro User Show in London and bought several add-ons for my computer – Starword and Starstore II to match my Cumana disc drive. I also took their advice as to which printer to buy.

After your review of SEDFS I ordered one from Slogger and received it by return. Minor problems (probably through my own lack of computing experience) ensued, but were quickly and efficiently dealt with by telephone. It is heartening in these hi-tech days to find a company where the customer is dealt with on such a personal basis. Thank you Slogger.

I find Electron User an excellent magazine, the content and structure is just about right. Why not a bi-weekly edition? Sorry to come across as a crawler, but when I am impressed I think it's worth saying so. – **A. Murray, West Ruslip, Middlesex.**

● The editorial team is already working flat out producing a monthly magazine. Some of them are cracking on a bit and the strain of a bi-weekly might be too much for their poor old hearts.

Disc driver

I HAVE an Acorn Electron fitted with a Rombox and Cumana disc interface connected to a Cumana disc drive and it works fine. But my discs do not work at school.

They have a BBC Micro with a Cumana disc drive. Is there a program or a chip that will let me use discs on a BBC Micro and Electron? I keep getting "Disc fault" on both micros.

If one day I can afford to get a bigger computer and change my Electron for a BBC Master, will that have disc problems? –**A. McPherson, Basildon, Essex.**

● There are two possible causes of the disc fault. Firstly, the Cumana interface on your Electron uses a completely different

---

ALL programs printed in this issue are exact reproduction of listings taken from running programs which have been thoroughly tested.

However on the very rare occasions that mistakes may occur corrections will be published as a matter of urgency. Should you encounter error messages when you type in a program they will almost certainly be the result of your own typing mistakes.

Unfortunately we can no longer answer personal programming queries concerning these mistakes. Of course letters about suggested errors will be investigated without delay, but any replies found necessary will only appear in the mail pages.

---

format disc to the BBC Micro.

You must use the utility disc supplied with the interface to format an Acorn DFS disc and copy your programs on it. Both the BBC Micro and Electron will then be able to read the files.

In addition to this you must use either 40 track drives on both micros or 80 track. You can't use 40 on one and 80 on the other.

Disc compatibility

I READ with interest the review by Mark Smiddy of the Slogger SEDFS for the Cumana disc interface in the June issue of Electron User. I was the envy of friends when I bought the original Cumana double density system.

It works very well and has many advantages over the standard Acorn DFS. The only drawback, as Mark says, is that there is no compatibility with the Acorn DFS used in BBC Micros and I found this an increasing irritation.

Some Acorn DFS discs would not copy to Cumana format and a few that did copy would not run properly. So, when I saw the Slogger SEDFS advertised I rushed out and bought one.

Yes, it is Acorn compatible, and yes, it is faster than the Cumana DFS, although speed is not always important. There is a good selection of utility commands. But there are some irritating drawbacks which caused headaches.

I find that ACP's Advanced

Disc Toolkit is very useful, but there are some commands which are not compatible with the Cumana DFS. Some are not compatible with SEDFS either, to the extent of leaving the drive motor running.

This problem has been corrected with a later version which Slogger very kindly sent me. – **A. Anderson, Devizes, Wiltshire.**

● We haven't experienced any problems copying our own unprotected Acorn DFS discs to Cumana format. However, commercial software is copy protected and is unlikely to work.

The Cumana disc interface uses a rather unusual disc controller and SEDFS must try to emulate the old 8271 chip found in the original BBC model B.

The emulation isn't quite perfect, though it is very good. We only found one or two utilities (such as a disc editor) that didn't work.

New chips for old

HAVING read your excellent magazine, I find your pages extremely interesting. I have only just bought an Acorn Electron, and I find it much better than other computers.

But I need help! I have an Electron with OS 1.0 and I am informed by my friends that I should get it upgraded to a 1.2. So, I set off for my local computer shop, but they didn't have one. Could you please tell me where I could get one from?

Also, how do you remove the old chip and replace it

---

with the new one? Is it just like the BBC Micro where a chip goes in and out of little sockets? And what are the advantages of OS 1.2?

I have a command which will perform a BREAK within a program. It is:

```
CALL !-4
```

Thank you for an excellent magazine which I will continue to buy and enjoy. – **Kenneth Rayner (13), Folkestone, Kent.**

● This is a question we've answered before but we still receive many inquiries. The BBC Micro's operating system has progressed from OS 0.1 when it was first launched to OS 5.1 in the Master Compact.

The Electron though has only ever had one operating system and that's OS 1.0. There aren't any other versions available and you can't use one from a BBC Micro or Compact.

Lost in shadow

I HAVE recently bought an Electron with Slogger's Master Ram Board and have been delighted with the additional memory available.

This has been particularly useful in compensating for the losses to the Plus 3, but there is one drawback. Both my four year old son and a friend of ten times that age have been distressed by their inability to save images created with the Nidd Valley Anamouse and Mousepaint program.

The rather brief documentation enclosed with the Master Ram Board suggests that the relevent memory can be accessed by machine code, but this, alas, is beyond my limited grasp of the subject.

Surely one of your staff can devise a solution to this problem. This would have fairly general application for users of Slogger's board who wish to take advantage of the space to run bigger graphics programs.

Approaching the subject from a different angle, I

---

have found a compaction utility which can effectively shorten Basic programs by stripping spaces and REMs, and by shortening variable and procedure names.

Can anyone take this a step further by devising a utility which will substitute the approved abreviations for keywords and/or replace the appropriate commands with VDU codes? – **Adrian Trout, Up Holland, Lancs.**

● The following program shows how to save a screen when using 64k shadow ram mode with Slogger's Master Ram Board:

```
10 REM Read/Write
20 REM Shadow memory
30 MODE 6
40 FOR i=0 TO 2 STEP 2
50 PX=&C00
60 COPT i
70 .read
80 PHA
90 LDA #0
100 BEQ here
110 .write
120 PHA
130 LDA #&40
140 .here
150 PHA
160 PLP
170 PLA
180 JMP &FBFD
190 ]
200 NEXT
210
220 REM Example screen
230 FOR i=1 TO 20
240 PRINT"Hello there..."
250 NEXT
260
270 REM Save it
280 file=OPENOUT"Screen"
290 FOR XX=&6000 TO &7FFF
300 YX=XX DIV 256
310 BPUT #file,USR read
320 NEXT
330 CLOSE #file
340
350 REM ReLoad it
360 MODE 6
370 file=OPENIN"Screen"
380 FOR XX=&6000 TO &7FFF
390 YX=XX DIV 256
400 AX=BGET #file
410 CALL write
420 NEXT
430 CLOSE #file
```

There is an extra operating system call to read and write to the screen memory. Call &FBFD with the address in X and Y and the overflow flag clear to read a byte into A, and set to write the byte in A.

All Basic keywords are stored in memory as single byte tokens, not as Ascii codes. For instance, COLOUR is stored as the value &90, so is COL. and C.

Using memory efficiently

WHEN are Advanced Computer Products bringing out their 256k ram cartridge that plugs into the Electron's cartridge port? As I write my own programs I am worried that as they get longer and more complex I will eventually run out of memory.

I don't want to buy another computer with a larger memory, as I think the Electron is an excellent machine suitable for my needs.

If the ram cartridge is produced, will I be able to use it with the Advanced Plus 4? Finally, will there be a 256k ram board for the Electron? – **Carolyn Crossley, Mitcham, Surrey.**

● ACP's ram cartridge will not expand the memory available to Basic programs. However, machine code programmers will be able to make use of it. It will work with ACP's Plus 4.

If you need more memory you could add either a second processor or Slogger Master Ram Board. These aren't cheap and there is a simpler, software solution. Steve Turnbull's Overlays article on page ● shows how to use the Electron's memory more efficiently.

– the abreviations. When a program is listed on the screen or printer, Basic does not print the token value, but prints the corresponding word instead. So, you can't shorten the keywords any further.

Synchron attack tips

IN SUPERIOR Software's Synchron I have almost blown up the enemy's HQ buildings and have some tips for this game:

If enemy space craft come on the screen, don't try to aim at them, they are too fast. Just fire rapidly across the screen until they go away.

Before you start to blast the orb containers, check that there are no defence pylons or force fields behind or in front of them.

When looking for the HQ or more orb containers try to follow the paths or roads. These are the safest routes, but don't stick too closely to them because there are many force fields, missile launchers and defence pylons next to them.

When you have collected all the orbs on level A, go down the screen for a while and you will find two big blocks of land targets. If you bomb most of them it will boost your score to over 10,000 points.

The distance to the HQ is shorter if you go up the screen, but you will also eventually reach it if you go

down the screen.

If you get a lot of orbs or have been exploring for quite a time, land at a runway because on your next life you will always take off on the runway you last landed.

I hope these tips will help people enjoy the game as much as I do. It is fast, but after a while you get the hang of it. – **Simon Hotchkiss, Shrewsbury, Shropshire.**

More power

I have an Electron, but unfortunately I can't use it because my adapter has broken. I have tried most of my local stockists without luck. Where can I get one as my programming powers are draining away – **Colin Anderson, Well Road, Lunanhead, Angus, Scotland**

● When you buy a Plus 3 you receive a new power supply adapter and the old one is redundant so there must be thousands gathering dust on shelves and in cupboards. Can any reader help Colin with a replacement?

Four player Dunjunz

I WOULD like to congratulate Bug Byte for producing Dunjunz on the Electron. It is an excellent and frustrating game.

With help from my friends, Courtenay Anderson and Timothy Lee, we have managed to complete the game by finding the chalice.

We did this by choosing the four player option. Then when the screens come up, one player goes off and tries to complete it.

If that person gets killed one of the other characters goes to find a cross to bring back the dead one. While this is happening another is guarding the ones left.

Playing the game this way saves the energy of the other characters so you can get further than you could if you went off by yourself. – **Chris Panter, Tavistock, Devon.**

Old timers drive slowly

CAN you publish some articles about the Plus 3's library suite of programs. A regular article about discs would be much appreciated.

Can you tell me why it is necessary to slow down the disc drive head as I have found no use for it so far. There must be some reason because this option is included in the utility SetParams. – **Peter Davey, Reading, Berks.**

● Some old disc drives may require a slow head stepping rate.

The Plus 3 drive is quite happy with a fast rate and shouldn't need altering.

If you want to start doing more with your micro than just playing games, this package is your ideal introduction to the four most popular applications for professional computers. All the programs have been designed for simplicity, so even a child can use them. Yet they include advanced features not yet available on programs costing many times as much!

Finalist for the Home Software of the Year Award

BRITISH MICRO COMPUTING AWARDS 1985

Contains 32 page booklet giving clear, easy-to-follow instructions on all 4 programs

Word Processor: Ideal for writing letters and reports. There is a constant display of both time and word count, plus a words-per-minute display to encourage the budding typist! A unique feature is the double-size text option in both edit and printer mode — perfect for young children and people with poor vision.

Database: You use this for storing information, just like an office filing cabinet. Facts you have entered can be quickly retrieved by just keying in a word or part of a word. They can be sorted, replaced, saved for future use or printed out.

Spreadsheet: Enables you to use your micro for home accounts or pocket money records. It creates a display of numbers in rows and columns. Continuous updating is possible, and a changed figure can be instantly reflected throughout the rest of the spreadsheet. Your results can be saved, to be used for future updates, or can be fed into its associated program . .

Graphics: Part of the spreadsheet section, it lets you draw bar charts, pie charts and histograms to give a graphic presentation of your statistics. Helps to give life and colour to the dullest figures!

☆ Word Processor ☆ Spreadsheet
☆ Database ☆ Graphics

Now they're all together – in ONE simple package

Four full-scale programs for an incredible low, low price! **£5.95** cassette

Valid to August 31, 1987

All prices include postage, packing and VAT
Overseas orders sent exclusively by Air Mail   £   p

Please enter number required in box   £   p

Electron User subscription
UK £12	3001	
Europe & Eire £20	3003	
Overseas £35	3004	

Commence with —————— issue

Renewal
UK £12	3002	
Europe & Eire £20	3476	
Overseas £35	3477	

The Adventure Creator
	With sub*	Without sub	
Cassette only	£8.95	£11.95	3122/3123

* This offer only available if accompanied by a subscription order or renewal

Computer Hits 3
	With sub*	Without sub	
Cassette only	£3.95	£6.95	3117/3118

* This offer only available if accompanied by a subscription order or renewal

Five Star
	With sub*	Without sub	
Cassette only	£3.95	£6.95	3126/3127

* This offer only available if accompanied by a subscription order or renewal

Repton 3
	With sub*	Without sub	
Cassette only	£4.95	£7.95	3009/3010

* This offer only available if accompanied by a subscription order or renewal

Electron User back issues
£1.50 UK		
£2.50 Europe	Jan 1987	3210
£3.50 Overseas	Feb 1987	3211
	Mar 1987	3212
	Apr 1987	3213
	May 1987	3214
	June 1987	3215
	July 1987	3216

Back Issue bundle **NEW**
April 86–Dec 86 (9 complete issues)   3017

UK £6.90, Europe inc Eire £8.90, Overseas £16.90

Cassette tape annual subscription
UK £35	3005	
Europe/Overseas £45		

Commence with —————— issue

Renewal
UK £35	3006	
Europe/Overseas £45		

Electron User tapes/discs
Tape £3.95, Discs £4.75. Overseas/Europe add £1

			Tape	3.5" Disc
Fishing	June 1986	3303		
Royal Wedding	July 1986	3304		
Howzat	Aug 1986	3305		
Cavern Capers	Sept 1986	3306		
Rat Catcher	Oct 1986	3307		3407
Day at the Races	Nov 1986	3308		3408
Santa's Sleigh	Dec 1986	3309		3409
Grid Warrior	Jan 1987	3310		3410
Maze	Feb 1987	3311		3411
Super Boss	Mar 1987	3312		3412
Dragons Doom	Apr 1987	3313		3413
Hectic Henry	May 1987	3314		3414
Hungry Harry	June 1987	3315		3415
Mazebugs	July 1987	3316		3416
Arena	Aug 1987	3317		3417

Add £2 for Europe, £5 for Overseas to all prices on this page unless otherwise indicated

Knitwear Designer **NEW**
Tape £8.95	3128	
5.25" Disc £9.95	3129	

Add £2 for Europe; £5 for Overseas

Nursery Rhymes **NEW**
	Tape £3.95	3.5" Disc £8.95
3018/3019		

Add £1 for Europe/Overseas

Education Special Vol. 2
Classroom Computing on the Electron

Magazine + Electron Cassette	£3.95	3007
Magazine + Electron 3.5" Disc	£4.95	3006

Add £2 for Europe £4 for Overseas

Fun School
		Tape £4.95	3.5" Disc £5.95
Ages 2-5	3080/3108		
Ages 5-8	3081/3109		
Ages 8-12	3082/3110		

Add £1 for Europe/Overseas

Getting Started in BBC Basic
UK only   £2.95   3100

Acornsoft Database
Needs a Plus 3   Disc £9.95   3119

Add £2 for Europe; £5 for Overseas

Advanced User Guide
UK only   £3.45   3072

Mini Office
Tape £5.95	3062	
Europe/Overseas £6.95		

Magic Sword
	Tape £5.95	3.5" Disc £6.95
3065/3105		

Add £2 for Europe; £5 for Overseas

Ten of the Best
		Tape £5.95	3.5" Disc £6.95
Volume 1	3068/3111		
Volume 2	3069/3112		
Volume 3	3070/3113		
Volume 4	3120/3121		

Add £1 for Europe/Overseas

French on the Run
Tape £9.95   3115

Add £1 for Europe/Overseas

Micro Olympics
Tape £5.95   3014

Add £1 for Europe/Overseas

Classic Card and Board Games
No. 1	Tape	£5.95	3090	
	3.5" Disc	£7.95	3092	
No. 2	Tape	£5.95	3091	
	3.5" Disc	£7.95	3093	

Add £1 for Europe/Overseas

Electron Dust cover
UK £3.95	3058	
Europe/Overseas £4.95		

Magazine Binder
UK £3.95	3059	
Europe £6.95		
Overseas £10.95		

➡ ➡ ➡ ➡ ➡ ➡ ➡ ➡   **TOTAL** ____

Payment please indicate method (✓)

☐ Access/Mastercard/Eurocard/Barclaycard/Visa

Expiry Date ___/___

No. ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

☐ Cheque/Eurocheque made payable to Database Publications Ltd.

Name ————————— Signed —————————

Address ————————————————

————————— Post Code —————

Tel —————————

Putting on the safety break

JAN EDENS presents a useful add-on to protect your Electron's Break key

HOW many Electron users can honestly say that they have never accidentally pressed Break while programming, or in a blind panic in the middle of a game?

It's particularly easy to do while using the cursor keys when editing a program listing.

At the very least it's an annoyance having to type OLD to recover your listing. At worst you could lose several hours of valuable data from your spreadsheet, or your best hi-score on galactic mega wars may be zapped, adding insult to injury when you have to reload the game.

This is just what the Anti Break System or ABS is designed to prevent. Not by a piece of inventive software, but by a simple hardware add-on that anyone can make at home for next to nothing.

The ABS is made from a small piece of 0.5mm aluminium. Figure I is a diagram of the ABS drawn full scale and may be cut out and used as a template.

After the metal has been cut, score and carefully bend the ABS along the lines indicated. Smooth the edges with a piece of emery cloth and when you've finished simply slide it into place over the Break key as shown in Figure II.

To use the Break key again simply slide the ABS out of the way.

This is such a simple, yet useful idea it's surprising it has never been thought of before. It really is true to say that "You'll wonder how you ever managed without it".

Figure I: Template for the Anti Break System

Figure II: The Anti Break System in place

Any port in a storm

Joe Pritchard shows how to make the most of the Plus 5's user port

LAST month we began our investigation of ACP's Plus 5 user port. We'll now look at it in greater detail.

First we need to look at the nature of the electrical signals that this I/O port recognises as representing zero and one.

Let's start by looking at what appears on a PB line when we are using it as an output. If we're outputting a one, then a signal of 5V appears and it's 0V for a zero.

However, if you were to measure this voltage with a good voltmeter, you'd probably find that the outputs weren't exactly 5V and 0V.

The manufacturers of the VIA in the user port simply guarantee that for a one output you'll always get a voltage above 2.5V and a zero output will always give a voltage of less than 0.4V.

Fortunately, the outputs actually obtained, unless you're unlucky, are usually quite close to 5V and 0V.

The 5V, though, can't provide us with much current and so can't do much work. It can't even light a small torch bulb.

For this reason, if you've got a line set up as an output you should never connect it to 0V as this could damage the VIA.

We can however, buffer the output signal from a PB line. This put simply, allows the PB line to control more current and so makes the output signal more useful. We'll look at one way of buffering the PB lines later on.

For a PB line set up as an input, any voltage above about 2.5V is treated as a one and any voltage below about 0.4V is regarded as a zero.

It's good practice to try to arrange our inputs, whether they're from switches or other electronic circuits, to be as close to 0V and 5V as possible. Never apply a voltage of more than 5V or less than 0V to a PB or CB line.

Electronic components whose input and output signals use 5V for one and 0V for zero are said to be

TTL compatible. TTL stands for Transistor Transistor Logic and it's the name of a vast family of logic chips in common use in most computers and peripherals.

TTL compatible means that the signals produced by the I/O port will be read as ones or zeros by TTL chips, and that TTL chips could provide inputs to the I/O port.

Let's take a closer look at the PB line as an input. Don't plug anything in to the user port, just enter and run the following program:

```
10 REM Program 5
20 ?&FC82=0
30 REPEAT
40 PRINT ?&FC80
50 UNTIL FALSE
```

You should find that the value 255 is printed all the time. This shows that all the inputs are one even though

we haven't got anything plugged in.

Don't worry, this is just a side effect of the VIA. The PB line always returns a one if read when set to input if nothing is connected to them.

How do we set about connecting something to the I/O port to provide an input? The simplest add-on is a switch and two are shown in Figure I.

The resistor on the left of the diagram is called a pull up resistor. When the switch is open it pulls up the PB line to 5V, thus giving a one input.

Although the PB line would give a one input when the switch was open anyway, the resistor just makes sure, which is good practice.

The resistor on the right in Figure I is called a pull down

resistor. When the switch is open, it pulls down the voltage on the PB input to 0V, thus giving a zero input.

In either of these circuits, don't be tempted to omit the resistor and replace it with a piece of wire. If you do, then close the switch, you'll short circuit the 5V line to 0V and crash the computer.

The 5V and 0V signals can, of course, be provided by other electronic circuits, such as another computer or peripheral. We'll look at some electronic circuits for providing inputs later in the series.

We've already seen that a buffer is needed before we can do anything useful with the I/O port outputs. The simplest buffer circuit is that shown in Figure II and uses a component called a transistor.
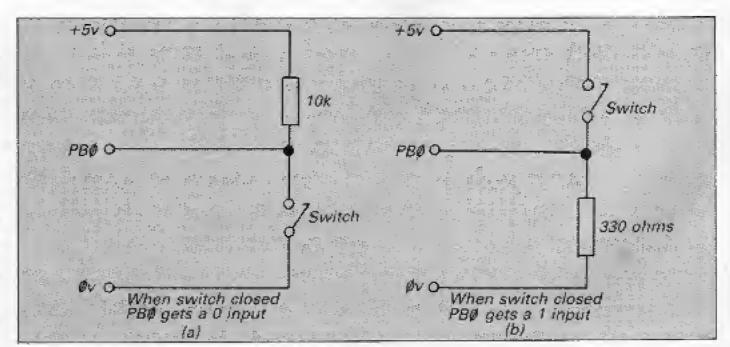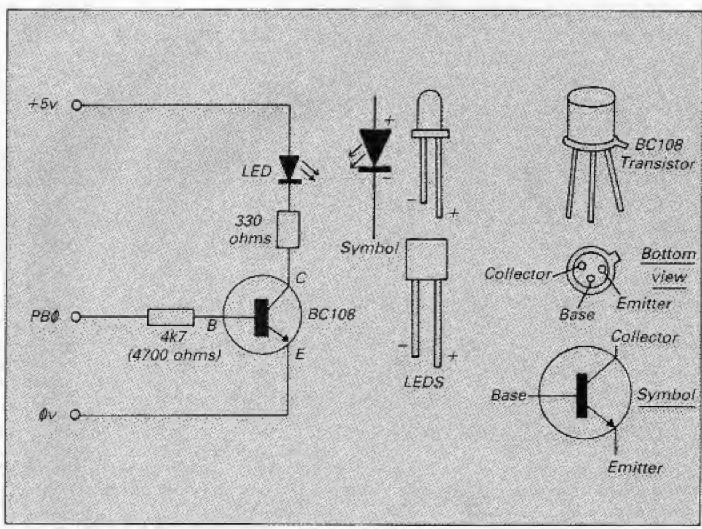
These come in thousands

Figure I: Two switches

[Figure caption text in diagram:]
+5v / 10k / PB0 / Switch / 0v / When switch closed PB0 gets a 0 input (a)

+5v / Switch / PB0 / 330 ohms / 0v / When switch closed PB0 gets a 1 input (b)

Figure II: A simple buffer circuit

You'll need a small piece of board with the holes 0.1in apart. Stripboard comes in large pieces, but you can cut it with a hacksaw and it's the sort of thing that we'll use quite often in later interfacing articles.

Before we start work, read the box on soldering transistors. It's crucial that the transistor and LED are soldered in the right way round.

Figure III shows the plain side of the stripboard. Spread the parts out so you've got plenty of room. It doesn't matter which strips on the board you connect the parts up to as long as you connect wires that are to be joined together to the same copper strip.

I suggest you solder the connecting wires and resistors in first, then the LED and finally the transistor. Again, take care to wire the transistor and LED in the right way around.

Position the resistors fairly close to the board when you solder them and cut off any spare wire. With the LED and transistor, leave their leads quite long.

Once the circuit is built we need to test it. First check that the transistor and LED are the right way round, and that there are no solder splashes or short circuits on the copper strips. If there are, remove them.

Turn off the Electron and plug in the terminal box that we built last month. Now
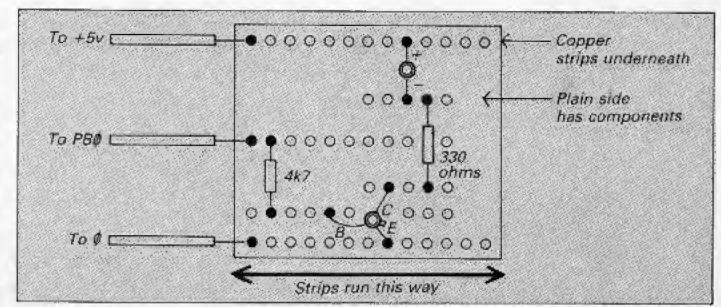
of different types and are designed for use in many different types of circuit, from transistor radios to computers. The transistor we'll use is called a BC108 and we'll connect it up as an electronic switch.

The other new component used in this circuit is a light emitting diode, or LED – the modern equivalent of a small torch bulb. You can get them in red, green and yellow.

How does this circuit work? Well, if the PB line is taken to 0V, then no current flows into the base connection of the transistor. This has the effect of turning the transistor switch off and no current flows through the LED.

If we take the PB line up to 5V by setting it to a one output, then a small current flows into the Base connection of the transistor. This turns the switch on and a fairly large current flows through the LED, resistor and transistor. This causes the LED to glow.

The voltage of 5V on the PB line causes a large current to flow through the LED

and a voltage of 0V causes no current to flow. The PB line thus switches current on or off.

The resistor between the LED and transistor is called a current limiter. It limits the current flowing through the LED and transistor to a safe level.

If you were to leave it out, such a large current might flow that the LED and transistor would be burned out.

The LED and resistor are called the transistor's load. The load in this circuit is simply the device that the

transistor passes current through. It needn't be an LED, it could be a motor, a light bulb or several other types of device.

Let's build this buffer. I built mine on a piece of stripboard. This is an insulating panel with holes in it which has copper strips on one side.

Components are pushed through the holes from the plain side and are soldered to the copper tracks which replace many of the wires we would normally need in an electronic circuit.

Figure III: The circuit built on stripboard

Hardware Projects

wire the 5V, 0V and PB0 line from this circuit to the terminal box.

There's nothing special about PB0. We could have used any of the other PB lines.

Now turn on the Electron. The usual start up message should appear. If it doesn't, turn off immediately and recheck everything, including the connections between the terminal box and the buffer circuit.

If the message appears, put your finger on the transistor. If it gets hot, you've got a problem, so turn off and recheck the circuit.

We now need the software. Any program using this buffer board must set the relevant bit of the DDRB to one to indicate that we want to use it as an output,

Soldering Transistors

Transistors, LEDs and chips all belong to a family of electronic components called semiconductors. They do all the work in electronic circuits and they are very delicate.

Heat, or connecting them up the wrong way around, can easily destroy them.

● Always check that the components are the correct way round before soldering them in.

● Try to solder semiconductors in place last. This limits the amount of heat they're exposed to.

● Take no more than 5 to 10 seconds to solder a semiconductor lead. Any longer than this and you stand a good chance of damaging it.

● Whenever possible, use a heat shunt. Pliers or tweezers will conduct some of the heat away from the delicate innards of the transistor. A third hand or a vice is useful to hold the circuit board.

● Keep the leads of transistors and LEDs as long as possible until you've got some experience under your belt.

then set the relevant bit of IORB to one or zero to turn the LED on or off.

If you've used PB0, then:
?&FCB2=1
will set up the DDRB. Program II will flash the LED on and off:

```
  10 REM User port demo
  20 REM By Joe Pritchard
  30 :
  40 ?&FCB2=1
  50 REPEAT
  60 ?&FCB0=1
  70 PROCdelay
  80 ?&FCB0=0
  90 PROCdelay
 100 UNTIL FALSE
 110 DEFPROCdelay
 120 TIME=0
 130 REPEAT
 140 UNTIL TIME>100
 150 ENDPROC
```

As an extension, you might like to write a program that reads the state of a switch connected to PB1 and starts the LED flashing if there is a one input to PB1.

● Next month we'll build a small buggy for your Electron to control. See you then.

Sidestep into space

STEVE TURNBULL shows how overlays can be used to cram more into ram

SQUEEZING quarts into pint pots is something that Electron programmers become very familiar with.

Even with all the possible ways of saving space (short variable names, leaving out REMs and spaces, multi-statement lines) there is still the 20k limit in Mode 6, and when you start adding graphics or disc drives the available space drops very rapidly.

Eventually, no matter how careful you are your Basic program will grow too big. What is needed is some way of sidestepping the space restriction.

One method is to CHAIN different programs to deal with different sections of the task being performed. But this technique suffers because only the Basic variables A%-Z% remain during the program change over.

The best method of getting round the problem is with overlays. An overlay is a section of Basic program which is loaded into memory when needed, and when not needed can be overwritten by a different overlay.

One of the techniques for merging two programs is to load the first program and *LOAD the second to TOP-2. Well, overlays uses the same technique.

All a program's procedures are stored on disc and whenever one is required it is loaded, tagged on the end of the program, called then deleted afterwards. So at any one time you've only got one procedure in memory making the program extremely short.

The advantages of over-

lays are that you don't lose any variables and you can have any number of them, subject to the number of files your filing system allows — 47 with a standard ADFS, with one file being used for an overall control program.

The disadvantages are that overlays are a little more complicated to set up and are harder to debug. You can use overlays with tape systems, but of course the loading speed slows things up too much.

First, we need to look at the usual memory layout of a typical Basic program.

This is shown in Figure I.

The addresses of the various sections of the Basic memory map are held in the pseudo-variables HIMEM, LOMEM, PAGE and TOP.

These can be used in the same way as ordinary variables, but unless you are careful Basic can lose track of where your program is.

Sitting at PAGE is the Basic program itself and this extends up to TOP. By typing:

```
PRINT TOP-PAGE
```

you can see how long your program is in bytes. Next comes the Basic heap which

holds all the variable names, values and procedure addresses. This starts at LOMEM and you can find out the address of the top of the heap with:

```
PRINT !2 AND &FFFF
```

Whenever a new variable is defined or a new procedure is called, its name and value are added to the heap.

Following the heap is a gap of free ram which gets eaten into as the heap grows. When you get a 'No room' error it means there is

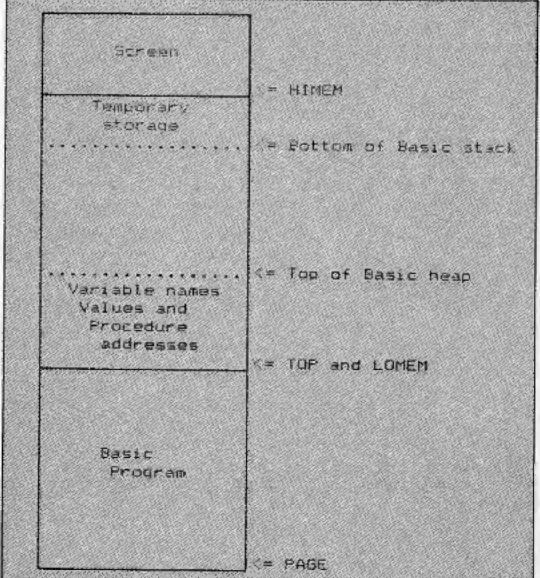Figure I: Usual memory arrangement of a Basic program

Screen	
	<= HIMEM
Temporary storage	
	<= Bottom of Basic stack
	<= Top of Basic heap
Variable names Values and Procedure addresses	
	<= TOP and LOMEM
Basic Program	
	<= PAGE

Programming

no free ram left and it is this error that we are trying to avoid.

Finally, just below the screen memory, sits the Basic stack. This is where Basic stores temporary values when it is doing calculations, plus other temporary information.

For overlays to be used we need a control program which is loaded first and stays in memory the whole time. When an overlay is needed it is loaded and tagged on to the end of the control program, effectively making it longer.

The problem now is: When the control program is loaded the values of TOP and LOMEM are set up as shown in Figure I.

Any variables used in the control program will be placed on the heap at LOMEM, if we then load an overlay on the end of the control program at TOP, these variables will get overwritten and lost.

But BBC Basic has already supplied us with the solution. Notice that there are two variables which are set to the same value: TOP and LOMEM.

By changing LOMEM to a different value we can reserve some space by
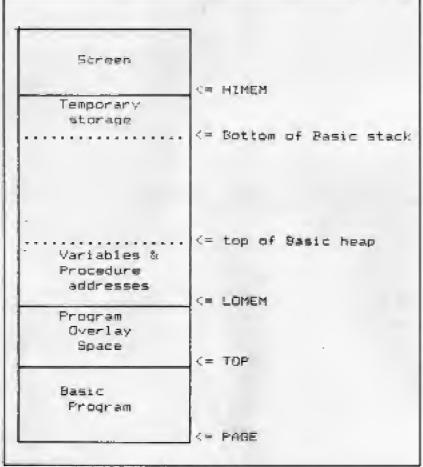
Figure II: Memory arrangement of a Basic program with overlays

changing the place where the heap starts. But we must do this before we define any variables or call any procedures.

Line 40 of Listing I creates a gap of 1k between TOP and LOMEM and this is the space the overlays will be loaded into. Figure II shows this new arrangement.

The next step is to organise the actual overlay loading. What we need is a procedure that can load the

overlays and merge them into the control program while the program is actually running.

At line 50 we set a variable to hold the address that the overlay will be loaded to and we write a procedure that takes a file name as a parameter and loads it as an overlay – unless it has already been loaded.

Type in Program I, the overlay control program and save it as *Control*, then clear the memory with NEW.

Now we need the overlays used in the control program. Each overlay looks like a Basic program, but only contains procedure definitions.

Type in Program II and save this as *One*. Clear the memory again with NEW and enter Program III, saving this as *Two*.

The first thing to notice is

that the two overlays can have the same line numbers – as long as you don't use GOTOs, GOSUBs and RESTORE line number, it doesn't matter what line numbers are used.

Next it is very good practice to prefix each procedure or function name with a couple of characters which are unique to that overlay, in this case 1– and 2–.

This protects against accidentally using the same procedure name in two different overlays which can be fatal because of the way BBC Basic works.
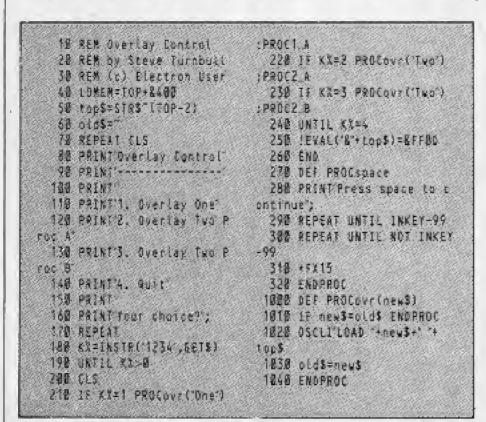
Each procedure calls PROCspace which sits in the control program – any routine which will be used by more than one overlay must be in the control program. Any routine which will only be used by one overlay should reside in that overlay to save space.

You are now ready to try out the overlays so CHAIN the control program. As you select the menu items the overlay needed is loaded unless it is already in memory.

When the quit option is selected the last overlay is removed by setting the old TOP of the program again.
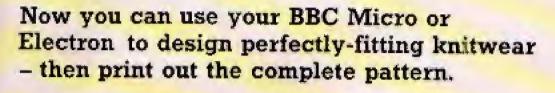
This is done because if you leave the overlay in place, modify the control program and then SAVE it, the new version will also contain the overlay.

And there it is, a useful way of getting around the memory limitations of the Electron and extending the size and power of your Basic programs. ■
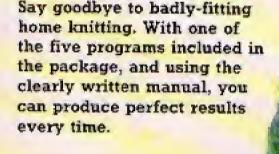
```
10 REM Overlay Control      :PROC1_A
20 REM by Steve Turnbull     220 IF K%=2 PROCovr("Two")
30 REM (c) Electron User     :PROC2_A
40 LOMEM=TOP+&400            230 IF K%=3 PROCovr("Two")
50 top$=STR$"(TOP-2)         :PROC2_B
60 old$=""                   240 UNTIL K%=4
70 REPEAT CLS                250 !EVAL("&"+top$)=&FFBD
80 PRINT"Overlay Control"    260 END
90 PRINT"--------------"     270 DEF PROCspace
100 PRINT"                   280 PRINT"Press space to c
110 PRINT"1. Overlay One"   ontinue";
120 PRINT"2. Overlay Two P   290 REPEAT UNTIL INKEY-99
roc A"                       300 REPEAT UNTIL NOT INKEY
130 PRINT"3. Overlay Two P  -99
roc B"                       310 *FX15
140 PRINT"4. Quit"           320 ENDPROC
150 PRINT                    1000 DEF PROCovr(new$)
160 PRINT"Your choice?"      1010 IF new$=old$ ENDPROC
170 REPEAT                   1020 OSCLI"LOAD "+new$+" "+
180 K%=INSTR("1234",GET$)   top$
190 UNTIL K%>0               1030 old$=new$
200 CLS                      1040 ENDPROC
210 IF K%=1 PROCovr("One")
```

Program I

```
5000 REM Overlay One
5010
5020 DEF PROC1_A
5030 FOR i=1 TO 10
5040 PRINT"Overlay One"
5050 NEXT
5060 PROCspace
5070 ENDPROC
```

Program II

```
5000 REM Overlay Two
5010
5020 DEF PROC2_A
5030 PROCprint("A")
5040 PROCspace
5050 ENDPROC
5060
5070 DEF PROC2_B
5080 PROCprint("B")
5090 PROCspace
5100 ENDPROC
5110
5120 DEF PROCprint(proc$)
5130 PRINT"Entering PROC2:"
proc$
5140 ENDPROC
```

Program III

Get knitting on your micro with
Knitwear Designer

Now you can use your BBC Micro or Electron to design perfectly-fitting knitwear – then print out the complete pattern.

It's the biggest development since the home knitting machine – and it's so simple to use. Now anyone can use a computer to design knitwear to individual measurements and then produce row-by-row knitting instructions on the printer.

YOU CAN:

- ● Work with any yarn, from fine cottons to chunky wools
- ● Enter your natural tension and the pattern adjusts to the way you knit
- ● Select from three preset patterns – Raglan sleeve, drop and square shoulder
- ● Use the Freehand program to produce custom designs

Say goodbye to badly-fitting home knitting. With one of the five programs included in the package, and using the clearly written manual, you can produce perfect results every time.

To order please turn to the form on Page 133

£8.95
Cassette
Also available on disc for use on the Advanced Plus 4 and Slogger SEDFS.
£9.95
5¼" disc

DATABASE PUBLICATIONS

Around the World in 40 Screens

REPTON

THE ARCTIC

AMERICA

THE ORIENT

THE OCEANS

AFRICA

Repton Returns . . . to Conquer the World

After the success of Repton 3, we were overwhelmed with requests for us to publish another set of screens for the game. We have now produced 40 new screens which, although enticingly amusing, are even trickier to complete than the original 24 screens of Repton 3.

These new screens feature Repton voyaging around the world. In America, Repton is a sheriff with a penchant for hamburgers and whisky; his job is to jail all the bandits and avenge the Red Indians. The Arctic sees Repton as an eskimo being "hotly" pursued home to his igloo by polar bears and penguins. Across to the Orient, and Repton is a kung-fu master amongst the pagodas, rickshaws and Chinese dragons. Repton becomes a deep-sea diver in the Oceans; he searches for pearls around the rotting shipwrecks which are inhabited by dangerous octopuses and squids. Finally, in Africa, Repton reaps his rewards by defeating the tribesmen, caging the parrots, and collecting the apples, bananas and pineapples — whilst watching out for falling coconuts!

THE SCREEN EDITOR

Each copy of Around The World in 40 Screens includes:
- the Repton 3 main program,
- the Repton 3 editor, and
- the 40 new game screens.

BBC Micro Cassette ....£6.95   Acorn Electron Cassette ..£6.95
BBC Micro 5¼" Disc £7.95
BBC Master Compact 3½" Disc £9.95

(Compatible with the BBC B, B+ and Master Series computers).
The screen pictures show the BBC Micro version of the game.

THE CHARACTER EDITOR

PRIZE COMPETITION

If you complete all 40 screens of Around The World, you can enter our competition. Prizes include £200 in cash, with Repton mugs, badges, pens and certificates for runners-up. Closing Date: 31st January, 1988.

FREE REPTON BADGE WITH EVERY GAME SOLD

A colourful Repton badge is included free with every cassette or disc of Around The World in 40 Screens.